School of Science
Master's Programme in Computer Science

Arto Suvitie

Control of an Autonomous Multicopter Fleet for Search and Rescue Missions

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science (Technology)

Espoo, January 29, 2018

Supervisor: Petri Vuorimaa, Professor

Thesis advisor: Enrique Ramírez, M.Sc.

AALTO UNIVERSITY
SCHOOL OF SCIENCE

ABSTRACT OF THE
MASTERS THESIS

Author: Arto Suvitie

Title of the thesis: Control of an Autonomous Multicopter Fleet for Search and Rescue Missions

Number of pages: 8 + 79 | Date: 29.01.2018

Major: Computer Science

Supervisor: Petri Vuorimaa

Thesis advisors: Enrique Ramírez

This thesis presents the requirements and implementation of a Ground Control Station (GCS) application for controlling a fleet of multicopters to perform a Search And Rescue (SAR) mission. The requirements are put together by analysing existing drone types, SAR practices, and available GCS applications. Multicopters are found to be the most feasible drone to use for the SAR use case because of their maneuverability, despite not having the best endurance. Several existing area coverage methods are presented and their usefulness is analyzed for SAR scenarios where different amounts of prior knowledge is available. It is stated that most search patterns can be used with a fleet of drones, by creating drone formations and by dividing the target area into sub-areas. It is noted that most currently available GCS applications are focused on controlling a single drone for either industrial or hobby use.

A proof of concept prototype is developed on top of an open source GCS and tested in field tests. Based on all the previous learnings from the protype and research, a new GCS is designed and developed. The development on optimizing communications between the GCS and the autopilot leads to a filed patent application. The new software is tested with three multicopters in a water rescue scenario and several user interface improvements are made as a result of the learnings. The development of a GCS for controlling a drone fleet for search and rescue is proven feasible.

Keywords:
UAV, Drone, Fleet, Swarm, Control, Search, Rescue

Publishing language: English

AALTO-YLIOPISTO
PERUSTIETEIDEN KORKEAKOULU

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä: Arto Suvitie	
Työn nimi: Autonomisen multikopteriparven hallinta etsintä- ja pelastustehtävissä	
Sivumäärä: 8 + 79	Päivämäärä: 29.01.2018
Pääaine: Tietotekniikka	
Työn valvoja: Petri Vuorimaa	
Työn ohjaaja: Enrique Ramírez	

Työssä esitetään multikopteriparven hallintaan käytettävän Ground Control Station (GCS) ohjelmiston vaatimukset ja toteutus Search And Rescue (SAR) etsintä- ja pelastustehtävien suorittamiseksi. Vaatimukset kootaan yhteen analysoimalla saatavilla olevia droonityyppejä, SAR pelastuskäytäntöjä, sekä GCS ohjelmistoja. Multikopterit osoittautuvat liikkuvuutensa ansiosta pelastustehtäviin sopivimmaksi vaihtoehdoksi, vaikka niiden saavutettavissa oleva lentoaika ei ole parhaimmasta päästä. Erilaisia etsintämetodeja esitetään alueiden kattamiseksi ja niiden hyödyllisyyttä analysoidaan SAR tilanteissa, joissa ennakkotietoa on saatavilla vaihtelevasti. Osoitetaan, että useimpia etsintäalgoritmeja voidaan hyödyntää drooniparvella, muodostamalla lentomuodostelmia, sekä jakamalla kohdealue pienempiin osa-alueisiin. Huomataan, että suurin osa tällä hetkellä saatavilla olevista GCS ohjelmistoista on suunnattu teollisuuden tai harrastelijoiden käyttöön, pääasiassa yksittäisen droonin hallintaan.

Prototyyppi kehitetään avoimen lähdekoodin GCS ohjelmiston pohjalta ja testataan kenttätesteissä. Tästä saadun tiedon avulla suunnitellaan ja kehitetään uusi GCS ohjelmisto. Kehitystyö viestinnän optimoinniksi autopilotin ja GCS ohjelmiston välillä johtaa patenttihakemukseen. Uusi ohjelmisto testataan kolmella multikopterilla vesipelastustilanteessa ja sen seurauksena käyttöliittymään tehdään useita parannuksia. GCS ohjelmiston luominen drooniparven hallintaan etsintä- ja pelastustehtävissä todetaan mahdolliseksi.

Avainsanat: UAV, drooni, parvi, hallinta, etsintä, pelastus	Kieli: Englanti

Contents

List of Figures

List of Tables

List of Abbreviations

2D	Two-dimensional
3D	Three-dimensional
EASA	European Aviation Safety Agency
EKF	Extended Kalman Filter
eNB	eNodeB Basestation
EPC	Evolved Packet Core
E-UTRAN	Evolved UMTS Terrestrial Radio Access Network
FAA	Federal Aviation Administration
GCS	Ground Control Station
GUI	Graphical User Interface

HD	High Definition
HSS	Home Subscriber Server
ICAO	International Civil Aviation Organization
IPP	Informative Path Planning
JARUS	Joint Authorities for the Rulemaking of Unmanned Systems
LiPo	Lithium Polymer Battery
Li-ion	Lithium-ion Battery
LTE	Long Term Evolution
MME	Mobility Management Entity
NBC	Nuclear, Biological and Chemical
PDN	Packet Data Network
PGW	Packet Data Network Gateway
SAR	Search and Rescue
SGW	Serving Gateway
TCP	Transmission Control Protocol
UAV	Unmanned Aerial Vehicle
UAS	Unmanned Aircraft System
UDP	User Datagram Protocol
UE	User Equipment
UI	User Interface
UTM	Unmanned Aircraft System Traffic Management
VTOL	Vertical Takeoff and Landing

1 Introduction

Drones, or Unmanned Aerial Vehicles (UAVs), were originally developed for military use during World War I as an evolution of aerial torpedoes (Keane, 2013). Since then, they have been adopted by hobbyists for recreational flights, photography and racing. In the last ten years, UAVs have also become increasingly prominent as professional tools for various civil and industrial use cases, such as agriculture, mapping, surveillance and inspection (Pastor, et al., 2007).

1.1 Motivation

UAVs are a versatile solution to a multitude of use cases thanks to all the different designs and payload options that are available. In area coverage based tasks, such as mapping or search, UAVs are cheaper to deploy and maintain than manned vehicles and they are faster than a group of people moving on foot. Also, when a task has to be performed in a hazardous environment, it is much safer to deploy UAVs than to send people who would have to put themselves at risk. UAVs, especially the multicopter variants, are also very maneuverable. This enables them to reach places that would be off limits for large, manned vehicles, due to their limited mobility in areas that have many obstacles. (Kohls, 2016)

UAVs can also be considered a solution that easily scales according to a given task and resources. Especially in scenarios involving coverage of large areas, the task can be carried out faster and more efficiently by assigning several UAVs to perform it cooperatively. This approach also has the added benefit of increased overall fault tolerance, as the mechanical failure of a single UAV does not compromise the whole operation. A group of UAVs cooperating to perform a task is often called a fleet or a swarm and there has been a lot of research into their use for various tasks.

Search And Rescue (SAR) use cases using a fleet of drones is very interesting. In contrast to, for example, area mapping, where it is sufficient to simply scan a given area with a rudimentary back and forth pattern, search and rescue tasks might not have a precisely defined area and the use of a coordinated fleet is much more important due to the time-critical nature of the task. It is also possible for the situation to change during the search mission because of additional information being discovered, in which case the drone operator must have the capability to react and change plans on the fly.

Despite the extensive research on the usage of drone fleets for SAR, there does not currently seem to be any software available that focuses on the control of a drone fleet for this task. Most drone control applications, also called by the term Ground Control Station (GCS), are focused on the control of a single drone. The goal of this thesis is thus the research, design and development of a GCS application, specifically built for controlling a fleet of several drones to carry out coordinated search missions to discover a missing person.

1.2 Objective and Scope

The scope of this thesis encompasses the design and development of a GCS application, which can be used to control a fleet of autonomous multirotor drones over a Long-Term Evolution (LTE) network to carry out a search operation for a missing person. As mentioned earlier, this thesis will focus on the development of the

control application. This also means that the details of the LTE connectivity will be out of scope for this thesis, because it has already been established in earlier research that LTE is a feasible choice for both controlling drones and for transmitting their payload video streams with a high bitrate (Sundqvist, 2015). Furthermore, to keep the scope manageable, the actual search methods will be low priority. Existing methods will be introduced and compared. The following items are considered out of scope for this thesis:

- The use of other vehicle types besides multicopters
- In-depth details of the drone's onboard computer
- The server implementation used to process payload streams from the drone
- Implementation of drone setup and calibration features to the GCS software

The literature review portion at the start of this thesis will provide the necessary background information needed to work on the main objectives. The review is split into unmanned aerial vehicles and the existing practices for search and rescue. The review of UAVs is meant to give an overview to the types of vehicles available and the capabilities that can be expected of them, which will be useful in determining which kind of vehicles should be used and evaluating their effectiveness in fleet operations. The SAR review will reveal what kinds of features need to be implemented during the development of a GCS application.

This thesis aims to answer the following two research questions:

- What requirements are there for a GCS application for controlling a UAV fleet in a SAR scenario?
- How to design and develop a GCS application that meets these requirements?

To find solutions to these questions, this thesis has the following four objectives.

The first objective of this thesis is to gain understanding of the state of the art of current drone control software. This will reveal what features exist currently, are thus possible to implement and will also give hints to what should be aimed for and what should be avoided when developing the new control application. The study will be done by finding various existing GCS applications, evaluating them and comparing them to each other.

The second objective is to define the requirements for a drone fleet search and rescue GCS application. This will be done by utilizing the information gained from the study of existing control software, as well as the literature review of SAR practices. To evaluate the feasibility and usefulness of the discovered requirements, a proof of concept prototype GCS application will be developed and tested. To get fast results, the prototype will be developed by using an existing open source GCS application as a base.

The third objective of this thesis is the development of a new complete GCS application from the ground up, based on the learnings from the proof of concept prototype.

The fourth objective is the evaluation of the final software in a realistic environment. This will be done by using the software in a practice SAR scenario, which will make it possible to evaluate how suitable the application is to the task and what remains to be improved upon.

1.3 Thesis Structure

This first chapter introduced the main topics, scope and objectives of this thesis. The second chapter begins the literary review, going into detail about unmanned aerial vehicles, their properties, classifications, control methods and some relevant legislation. Next, the third chapter continues the review by introducing existing practices for search and rescue use cases, focusing on search methods. After that, the fourth chapter introduces several existing GCS applications in order to find out what is the current state of the art. Next, in the fifth chapter the specifications for a SAR GCS application are listed and in chapter six a prototype is built and tested. Then, in the seventh chapter, a new GCS application is implemented based on the learnings from the prototype project and then tested in a field test. This chapter also mentions the communications optimizations that were made between the GCS and the autopilot that led to a filed patent application. Finally, the last chapter contains conclusions and analysis and presents proposals for future work.

2 Unmanned Aerial Vehicles

This chapter provides an overview to the types of UAVs available, their characteristics, use cases, payloads, control methods and relevant legislation.

2.1 Overview

UAVs, or drones, are remotely operated aircraft without an onboard pilot, which can be set to carry out autonomous flight missions. They come in several different types and categories, suited for different use cases and budgets. The frame of the UAV mainly determines the general architecture of the vehicle, while the classification determines the scale. As mentioned in the introduction, UAVs are relatively fast, maneuverable and cheap. However, they do have their own set of challenges as well. The most relevant of these being limited carrying capacity and flight time.

2.2 Use Cases

As mentioned earlier, UAVs are versatile tools fit for a multitude of use cases. They are especially useful to deploy in situations that are considered dangerous, dirty or dull for humans. Some examples of such scenarios include environmental research, pollution assessment, border monitoring, fishery applications and oceanography, relaying communications (Pastor, et al., 2007), (Bayat, et al., 2017). UAVs are also useful for monitoring wildlife populations (Hodgson, et al., 2017).

When it comes to SAR, drones can be deployed to perform search with High-Definition (HD) and thermal cameras. In the case of an environmental disaster, they can be used to create two-dimensional (2D) and three-dimensional (3D) maps of the area. These maps can be useful, for example, in post-earthquake assessment (Nedjati, et al., 2016). The drones can also be equipped with delivery systems that can be used to, for example, deliver automated external defibrillators as cardiac arrest response (Boutilier, et al., 2016). There are also cases where UAVs have been used to collect information that enabled damage assessment after natural disasters such as hurricanes, floods and earth quakes (Scott, 2016).

Furthermore, UAVs can be used to create 3D maps for agricultural assessment, archeological and architectural surveys, and monitoring of forestry, environment, excavation volume, contaminated areas and traffic (Nex & Remondino, 2014).

2.3 Frames

The frame defines the general structure of a UAV, which means that it has a major impact on the achievable performance and capabilities of the vehicle. There are many types of frames available and it is important to find the most suitable frame for the use case that the UAV will be deployed for. This chapter goes over the different frame types available, points out their strengths and weaknesses, and finds the most suitable types for the search and rescue use case.

The most common UAV frame types are fixed-wing and multicopter frames. A bit less common are single-rotor vehicles that resemble helicopters and Vertical Take-Off and Landing (VTOL) vehicles, which can be

considered as a hybrid between the fixed-wing and multicopter designs. Also, for niche use cases there exists frames like blimps and rockets.

Fixed-wing

Fixed-wing UAVs, as their name describes, use a wing as their main source to provide lift, which means that they only need to expend energy for forwards movement. As a result, they are very energy-efficient and capable of reaching very long flight times, many of them being able to stay in the air for well over 16 hours (Chapman, 2016). The long flight times make the fixed-wing design a very suitable choice for long range missions such as area mapping. On the other hand, the biggest drawback of this design is the relatively limited maneuverability. Fixed-wing UAVs cannot stop and hover in place and they are incapable of performing sharp turns. Furthermore, their takeoff and landing procedures require the use of runways, catapults, parachutes or nets, which complicates and slows down their deployment and recovery. Because of these limitations, this frame design is rather unsuitable for tasks that require quick deployment, precise movement, or capability for ad-hoc changes in the flight route such as inspection of specific locations. A fixed-wing UAV built for aerial mapping is shown in Figure 1 below.

Figure 1: A fixed-wing UAV (ArduPilot Dev Team, 2017)

Multicopter

Multicopters are UAVs equipped with three or more rotating propellers, with the most common configuration being quadcopter with four propellers. Compared to fixed wing vehicles, multicopters are significantly less energy-efficient due to their dependency on their propulsion to keep them in the air for the entire duration of the flight (Scott, 2016). A multicopter with six propellers is shown in Figure 2 below.

Figure 2: Hexacopter, a multicopter with six propellers. (ArduPilot Dev Team, 2017)

Multicopters usually have one rotor on each arm, but they can also be built in a coaxial rotor configuration, in which case each arm of the UAV is equipped with two propellers on top of each other, rotating in opposing directions. This increases mechanical complexity of the system, increasing the risk for mechanical faults, but it also provides more stability and flight time to the UAV. It is also worth noting that the coaxial rotors can easily get in camera's view, when the UAV is equipped with one.

Most multicopters can only stay in the air for roughly half an hour, although new power sources are constantly being developed to reach longer flight times. The main advantage of these vehicles is that they have the best available maneuverability, allowing them to hover in place, make sharp turns and takeoff and land vertically on a small landing area. This flexibility makes multicopters suitable for flight missions where the flight plan may need to be adjusted or paused on demand.

Single-rotor

Single-rotor UAVs are vehicles designed after traditional helicopters with a single rotor providing the lift and a small tail rotor taking care of the yaw-axis rotation. Their main advantages are that they are also capable of hovering in place unlike fixed-wing vehicles and that they are more efficient compared to multicopters because of their large, relatively slowly spinning propeller. However, the large propeller is also capable of causing more severe injuries, and the lack of additional rotors makes recovery of motor failure impossible. Also, single-rotors are more complex mechanically, which means that they are more expensive to purchase and require more maintenance. They are also prone to vibrations that can affect the quality of payload video streams. This makes them too risky for deployment on important search tasks, where reliability and video quality are required. One such UAV is shown in Figure 3 below.

Figure 3: Single-rotor UAV (ArduPilot Dev Team, 2017)

Vertical Takeoff and Landing

VTOL vehicles are hybrids between fixed-wing and multicopter designs. They have wings and propellers that allow them to get some of the maneuverability of quadcopters and some of the energy efficiency of fixed-wings. Some designs incorporate tilting rotors, which allow the VTOL vehicle to rotate the rotors that initially provide vertical lift to give forwards thrust during flight after takeoff. These designs are interesting, but they bring complexity, which can be considered as added potential points of failure. In future works, these types of vehicles should be kept in consideration, but for now they appear to be too risky to be used in tasks that require reliability. Figure 4 below depicts a VTOL UAV that has fixed rotors.

Figure 4: A quad plane VTOL UAV with fixed rotors (ArduPilot Dev Team, 2017)

Blimps and Rockets

Blimps and rockets are some of the less frequently seen types of UAV frames. Blimps are slow moving low altitude vehicles that can reach long flight times and they are best suited for observation tasks that can last for several hours (Scott, 2016). In addition to being slow, blimps are also vulnerable to weather conditions, which further limits their usability. Rockets on the other hand are very fast but also very limited in terms of maneuverability and flight times, which makes them unsuitable for performing the maneuvers required by even the most basic search patterns. Because of their limitations, neither of these frame types are useful for the use cases of this thesis. Examples of these UAVs are shown in Figure 5 below.

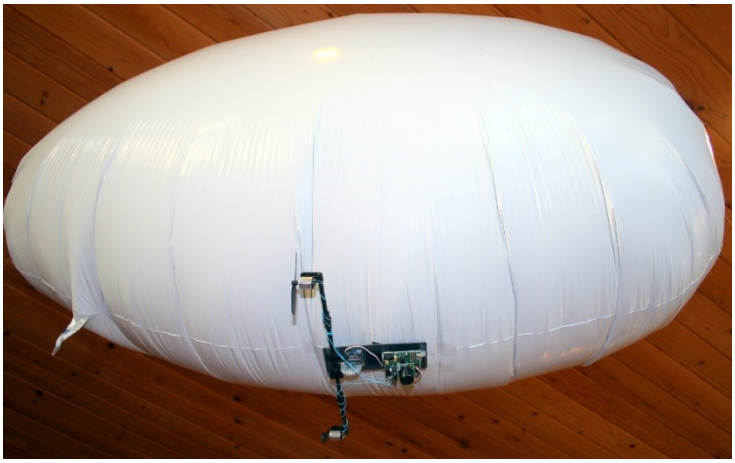

Figure 5: An autonomous blimp (Anderson, 2008) and a rocket UAV (Krupnik, 2012)

Summary

The strengths and weaknesses of the presented UAV frames are summarized in Table 1 below.

Table 1: UAV frame comparison

Frame type	***Pros***	***Cons***
Fixed-wing	Long flight time	Limited maneuverability
Multicopter	Good maneuverability	Limited flight time
Single-rotor	More efficient compared to multirotor	Mechanical complexity
VTOL	More endurance compared to multicopter More maneuverability compared to fixed-wing	Mechanical complexity
Blimp	Very long flight time	Slow Poor maneuverability Vulnerable to wind conditions
Rocket	Fast	Poor maneuverability

2.4 Power Sources

The power source plays a major role in the achievable flight time and carrying capacity of the UAV. Most UAVs are electrically powered and carry lithium polymer batteries. However, there are other notable power sources that have become more prominent recently.

Lithium Polymer

Currently, the most common power source for UAVs are lightweight lithium polymer (LiPo) batteries. Their main advantages are their high discharge rate and a form factor that easily fits in UAV frames (McCray, 2015). A typical LiPo battery is shown in Figure 6 below. Very small hobby drones are often equipped with 3-Cell LiPo batteries, which provide roughly 10 minutes of flight time. Larger drones can be fitted with 6-Cell LiPos

and they can generally fly for half an hour. There has been research and experiments on technologies for automatic battery replacement for LiPo equipped drones (Suzuki, et al., 2011).

Figure 6: LiPo battery (Lowc Technology Co., Ltd, 2017)

Lithium-Ion

Lithium-ion (Li-ion) batteries are very popular in consumer- and portable electronics. They have a cylindrical, rigid metal casing. Their advantage over LiPo batteries is that they have a significantly higher energy density, which directly means longer flight times for UAVs, despite the greater weight caused by their casing. They are also cheaper to manufacture and purchase. However, the low discharge rate of Li-ion batteries makes them unsuitable for UAVs. This is because sometimes a UAV needs to speed up a motor to stabilize the vehicle to prevent it from crashing. With low discharge rate power source, there will be not enough power available to perform the maneuver (McCray, 2015).

Figure 7: Li-ion batteries (Hong Kong TAC Industrial Co., Ltd., 2017)

Solar Panels

There have been attempts at fitting solar panels on drones to generate extra energy during flight. The main benefit of this method is that it enables the drone to passively generate energy. However, the amount of energy generated is low and the panels require free surface area on the drone. These aspects make the solar panels have very limited benefit especially on multicopters, which do not typically require a lot of energy and have limited surface area. Also, increasing the surface area of a multicopter to accommodate more solar panels increases the drag during flight, which negatively impacts the efficiency of the vehicle. This has been confirmed by hobbyists (Lukonis, 2014).

Still, solar panels can provide value for fixed-wing UAVs, whose energy consumption is very low during gliding and they tend to have surface area available across their wingspan. Solar-powered UAVs are shown in Figure 8 below.

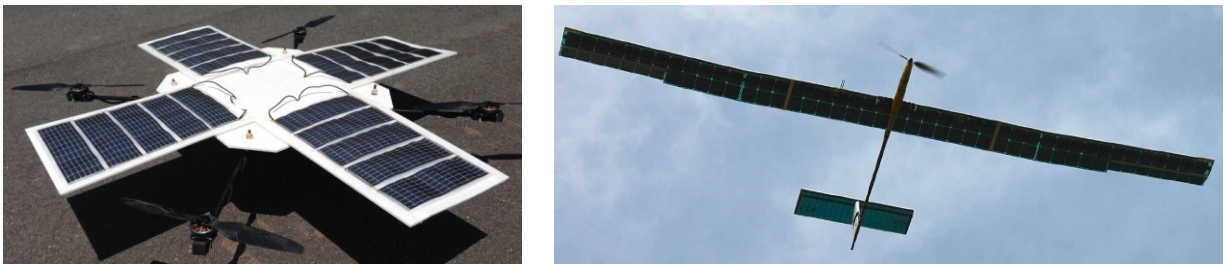

Figure 8: Solar panels fitted on a quadcopter (Lukonis, 2014) and on a fixed-wing UAV (Oettershagen, et al., 2016)

Gasoline

There are two-stroke gasoline engines that have low enough mass and high enough power output to be mounted on heavy duty UAVs as a power source (Hooper, 2005). These engines have the advantage of allowing the drone to stay in the air for hours, but on the other hand they produce a constant loud noise that could be compared to that of a chainsaw or a motorcycle. This makes the UAV unsuitable for carrying any sound sensitive payloads such as microphones or flying near people that could be bothered by the noise. The engine also produces considerable vibrations, which must be considered in the design of the drone so that it won't cause issues with video feeds. Also, the gasoline engine cannot be considered environmentally friendly, unless the drone is used as a substitute for a manned helicopter.

Internal combustion engines are the power source of choice for many state-of-the-art UAVs due to having roughly an order of magnitude greater power and energy densities in comparison to electrical motors. However, combustion engines are not as efficient, quiet, ecological, economical or reliable as their electrical counterparts. (Dudek, et al., 2013)

Figure 9: Liquid and Air-Cooled gasoline engines (Hooper, 2005)

There are also hybrid solutions, where the UAV carries both a gasoline engine and LiPo batteries. With this setup, the engine is used to charge the batteries, which in turn power the motors. A hybrid quadcopter capable of flying more than four hours is shown in Figure 10 below.

Figure 10: Hybrid gasoline-electric quadcopter (Quadcopter-Addiction.com, 2017)

Hydrogen

Hydrogen fuel cells are a rapidly developing power source technology. Figure 11 below shows a quadcopter that is carrying a hydrogen fuel tank under it and two fuel cells on top of it, enabling it to stay in the air for two hours (BBC, 2016). At the time of writing this thesis, there are some hydrogen powered drones that can stay in the air for hours and they are much quieter and more environmentally friendly than their gasoline counterparts. However, the fuel cells suffer from low power density, which results in these drones having a very limited carrying capacity and being very vulnerable to wind conditions (Tao, 2016). This means that the technology is currently not yet suitable for rugged SAR use cases. For future research, it is worth keeping an eye out for new developments with hydrogen power sources.

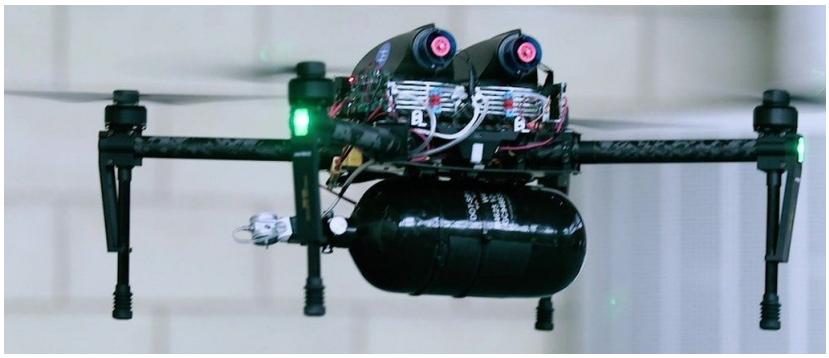

Figure 11: A quadcopter equipped with hydrogen fuel cells (BBC, 2016)

Summary

LiPo batteries, despite providing relatively low flight times, are still a preferable choice for drones. Hydrogen fuel cells are not quite there yet, so currently gasoline engines seem to be the best option for long flight time multicopters. Battery technology is advancing all the time, which means that the current greatest weakness of multicopters, their flight time, may become less limited in the future. A comparison of the presented power sources is shown in Table 2 below.

Table 2: Comparison of power sources

Power source	***Pros***	***Cons***
LiPo	High discharge rate. Convenient form factor and weight.	Limited power density.
Li-ion	Relatively high energy density.	Low discharge rate prevents UAV from drawing extra energy for maneuvers.
Solar panels	Generates electricity during operations.	Not efficient enough to provide value for a multicopter.
Gasoline	Provides long flight times.	Generates noise, vibrations and exhaust fumes. Not as reliable as electric counterparts.
Hydrogen	Environmentally friendly.	Low power density makes the UAV vulnerable to wind conditions.

2.5 Payloads

The main value of UAVs is provided by the payloads they carry. An important aspect of the payload is its weight, since it has a considerable effect on the UAVs endurance and operational lifetime, due to heavier payloads requiring more energy to maneuver (Scott, 2016). The size and dimensions of the payload are also important, because if the payload is too large it might get in the way of the propellers or extend beyond the drones landing gear.

Cameras

Cameras are a popular payload choice for drones. There are a few different types of cameras available.

Regular HD cameras are a typical choice. They can transmit a real-time video stream, which can be used to both survey the area and to help navigate the drone in its surroundings. They can also be used for mapping to construct an accurate 2D representation of the target area or object. Still images captured from these cameras can also be used to construct 3D models of the area. This will require the UAV to take several pictures from multiple angles of the object. The 3D models can be used for emergency management and rescue planning in disaster scenarios where the landscape has been deformed (Nex & Remondino, 2014).

Thermal cameras enable the detection of warm objects regardless of how dark it is. This is useful for the search and rescue use case, especially in cold environments, where people are easy to spot. However, if the environment is roughly the same temperature as the target that is being searched, then the thermal camera will not be of much use. In addition to the SAR use case, thermal cameras are also used in industrial surveys to find leakages in pipes and structures, for example.

Multispectral cameras can be used to see more than what can be seen with a regular HD camera or a human eye. Example use cases for such cameras are chlorophyll concentration detection (Zarco-Tejada, et al., 2009) and agronomical plant health assessment (Nebiker, et al., 2008).

A gimbal is needed for mounting a camera to a drone. The main benefit of using a gimbal is that it counteracts the shaking and rolling of the drone, keeping the camera stable. Gimbals also enable the camera to be rotated, which means that the camera can be pointed towards a desired target regardless of which direction the drone itself is heading. Mounting a camera to a drone without a gimbal will cause the video output to be very shaky and blurry.

Distance Sensors

There are sensor payloads that measure the distance from the drone to the first colliding object. These sensors can be used to implement collision avoidance on the drone and with the more elaborate sensors, even to perform 3D mapping of the area. Performing the 3D mapping using a lidar is much faster compared to the photogrammetry method discussed above, but it lacks the color information, as the data is all contained in a single large 3D point cloud.

Sonars are a rather cheap and lightweight distance sensor. However, their range is typically very low, falling in the ballpark of 10 meters, which is not going to be enough to prevent a collision for a fast-moving drone.

Laser rangefinders can detect longer distances. However, since they only measure a single point, they cannot be used as a reliable collision prevention tool. They are used to detect drone altitude by pointing them straight down. This will give a more accurate reading compared to the built-in barometer of the drone's autopilot.

Lidars work in a similar fashion to laser rangefinders, except that instead of returning the distance to a single point, they scan an area. This makes them more suited for collision prevention and also makes it possible to use them for 3D mapping of the environment.

Radiation and Gas Sensors

UAVs can be fitted with gas sensors to evaluate air quality and to detect chemical leakages. One of the use cases for such sensors is to have the UAV help prevent or mitigate acts of terrorism by detecting Nuclear, Biological and Chemical (NBC) threats (Aguilar & Benítez, 2004). There has also been research on path-finding algorithms for radiation-sensor equipped drones to locate radioactive hotspots quickly (Newaz, et al., 2016).

Actionable Payloads

The previously mentioned payloads have all been sensors of some type, which allow the UAV to monitor and evaluate its environment in some way. However, there are also payloads that enable the UAV to perform active interactions.

A delivery system enables the UAV to transport objects from one place to another. Typically, smaller objects can be fitted with parachutes and dropped from the air near the desired location. If the object is heavy or it needs to be delivered to a precise location, it is more reliable to have the drone land with the payload at the desired location, detach the payload, takeoff and fly back to home location.

There are also some payloads for direct interaction with people. For example, a loudspeaker can be mounted on a drone for giving announcements or instructions (Luke & Uchizono, 2011). Research has shown that verbal communication from a robot helps in SAR scenarios by decreasing fear and increasing cooperation (Groom, et al., 2011).

Service Providing Payloads

Small network base stations can be mounted on UAVs to provide temporary local cellular coverage to an area. This is useful whenever there is an unexpected or temporary need for connectivity, for example because of a natural disaster or a sudden extreme density of users in an area (Bor-Yaliniz, et al., 2016). In the natural disaster scenario, the aerial base station can be used to provide emergency communications capabilities to disaster areas, which can then be used by other UAVs and on-site personnel to relay information. A fleet of UAVs can also be used to form a relay network to provide blanket coverage for a wide region simultaneously (De Freitas, et al., 2010).

2.6 Wireless Communications

Wireless communications are used to send commands to the UAV and to receive telemetry and payload-data in real time.

Drones can be controlled wirelessly in real time by a pilot using a radio controller or ground control station software. However, it is also possible to set the drones to carry out predefined flight missions or to have the drone carry an onboard computer that handles navigation autonomously. In these cases, the drone can perform its tasks even in environments and situations where pilots or connectivity have limited availability.

Long-Term Evolution

In this thesis, 4G Long-Term Evolution (LTE) has been chosen as the primary communication method between the drones and the ground station.

The LTE network architecture has three main components: The User Equipment (UE), the Evolved UMTS Terrestrial Radio Access Network (E-UTRAN) and the Evolved Packet Core (EPC) (Astély, et al., 2009). An overview of this architecture is shown in Figure 12 below.

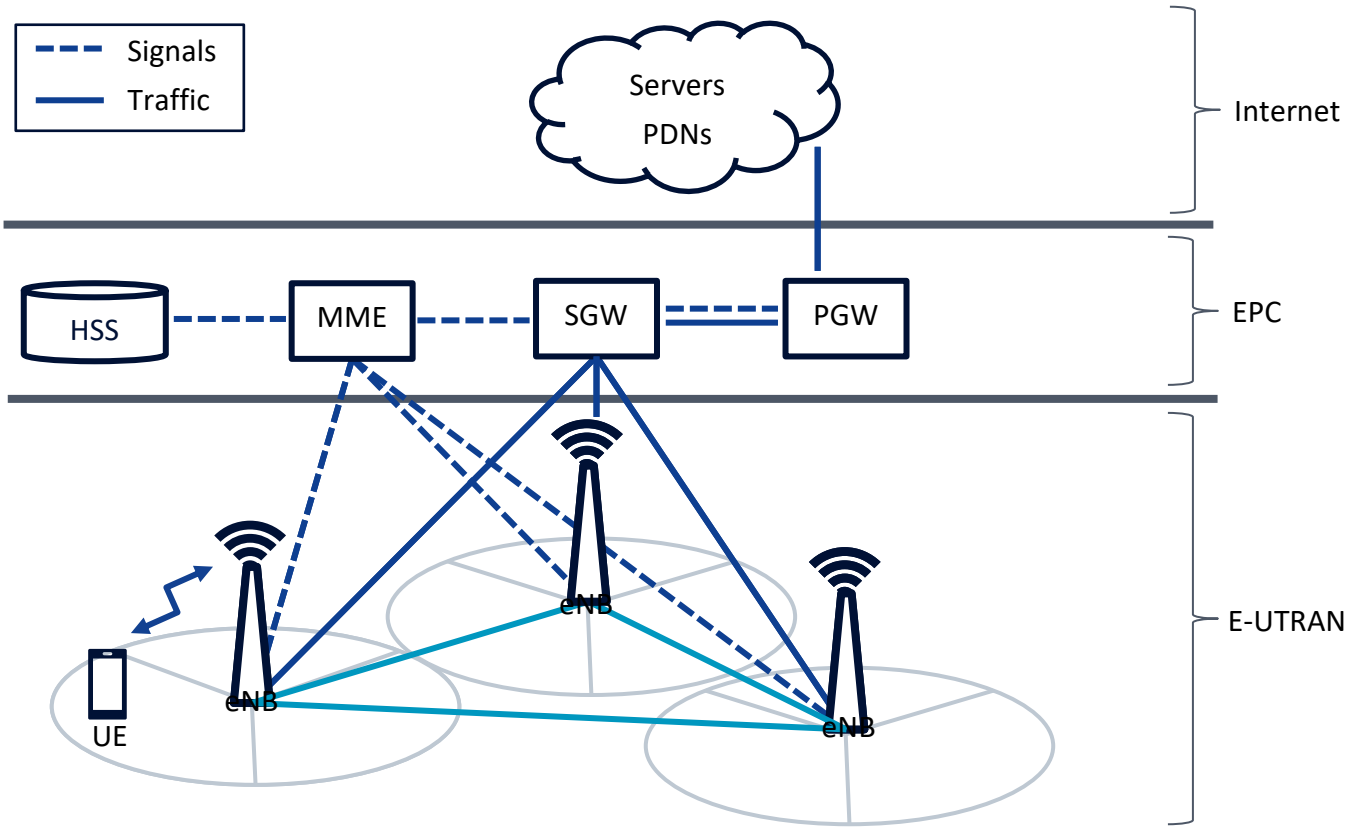

Figure 12: Architectural overview of the LTE network

The UE refers to any device that connects to the LTE network, typically a mobile phone or a laptop. In the use case of this thesis, relevant UEs are the ground station laptop and the drones themselves. An LTE modem with a registered SIM is required to enable a laptop or a drone to connect to an LTE network. The modems can often be connected to antennas for improved range.

The E-UTRAN, as the name implies, forms the access network, which comprises of multiple evolved base stations called eNodeB (eNB). These base stations serve one or more cells, which the UEs can connect to. In the figure above, these cells are shown as three sectors around each eNB. The main task of the eNB is to handle the communications between the UEs and the EPC. In addition to being connected to UEs and the EPC, each eNB is also connected to its nearby peers for the purpose of signaling and handover packet forwarding. Each UE can belong to only one cell and communicate with only one eNB at a time, and a handover must be performed whenever the UE moves to a new cell.

The EPC forms the core network and it contains the Home Subscriber Server (HSS), the Packet Data Network Gateway (PGW), the Serving Gateway (SGW) and the Mobility Management Entity (MME). The HSS is a central database for information related to users and subscriptions. It is queried by the MME, which is responsible for control plane operations and UE authentication. Each UE is assigned their own SGW, which handles the routing, forwarding and buffering of packets. PGW on the other hand handles IP address allocation for UEs and does all IP packet operations required for the connections towards the Packet Data Network (PDN).

One of the advantages of the LTE technology is the capability to use spatial multiplexing with Multiple Input Multiple Output (MIMO) technique. This means that both the sender and receiver use multiple antennas simultaneously to transfer multiple data streams, increasing the bandwidth of the link.

Drones are traditionally controlled through a direct radio link, where both the drone itself and the ground station or handheld controller are equipped with radio transceivers and antennas. In this case, the vehicle must stay within range of the ground station and within line of sight, or risk losing the connection. On the other hand, if an existing LTE network is used, then the vehicle can freely move wherever there is network coverage. This also means that the control software may be physically in a completely different location.

Real-time Teleoperation

Teleoperation refers to the traditional low-level control method where the drone is controlled with a radio controller or joysticks that allow for very precise maneuvering. However, this method requires skill and attention from the user, which prevents them from simultaneously performing other tasks (Szafir, et al., 2017). An alternative real-time control method, that demands less skill and attention, is guided waypoints. With this method, instead of controlling the drone directly, the user specifies a location to which the drone will directly attempt to move to. Typically, this is implemented as a map on a ground station application, where the user may click to set the current target waypoint for a drone. In comparison to joystick controls, the guided waypoints are less precise.

Autonomous control can be considered a more sophisticated control method, which differs from direct teleoperation by having an UAV capable of performing operations even when communications are lost for a period of time. In its simplest form, autonomous control can be implemented as a list of mission items, that the UAV will carry out in sequence. Here, the loss of communications does not necessarily mean that the mission will be aborted since the commands are all stored locally in the drone. However, depending on the situation, an unconnected drone can be a safety risk, in which case it should be set to pause the mission or return to launch upon losing connection. MAVLink based vehicles implement their autonomous missions in this fashion (Meier, 2017).

The commands mentioned above are sent directly to the autopilot of the drone, which is responsible for carrying out basic flight functionality. However, a more advanced form of autonomous control can be implemented by fitting the drone with an additional onboard computer, which will then be responsible of defining the mission details and performing adjustments to the flight plan in real time according to information received from sensors. In this scenario, the ground station would only give high level commands, leaving the details up to the vehicle. The onboard computer can also be used to perform collision avoidance or to communicate directly with other drones' onboard computers.

The methods mentioned earlier are not exclusive of each other and can be used together. The operator can use a ground control station to plan a waypoint-based mission, which can then be interrupted with direct teleoperation if, for example, something interesting is spotted from the drone's video stream, which requires

closer inspection. Concurrent to this, the onboard computer can monitor the drone's status in real time and act when necessary.

Even though teleoperation via joystick and waypoint mission planning and control can be all incorporated into a single ground station application, it is preferable to also have a radio controller available as an emergency backup option in case the primary connection or control method is lost.

Telemetry and Payload Data Reporting

During flight, the UAV sends a continuous stream of telemetry reports back to the ground station. These reports contain information about the location and status of the UAV, as well as acknowledgements for commands and updates for current mission status. The most important part of these reports are the battery charge and error status, which can be used to abort a flight before there is risk of crashing the UAV.

2.7 Cooperative Swarming

As mentioned in chapter 2.2 earlier, a single drone can be used for a multitude of use cases. However, the use of several drones makes it possible to complete tasks, such as area search much more efficiently. A group of drones (often referred to as a fleet or a swarm) can cover a larger area faster before having to replace the batteries.

It has been determined that using multiple miniature UAVs to perform tasks is cheaper in both acquisition and maintenance in comparison to using a single large UAV (Scott, 2016). Furthermore, a fleet of UAVs can be scaled up to provide better coverage and endurance, since the failure of a single drone will not compromise the entire mission.

Implementing a drone fleet solution brings its own set of challenges. The user must be able to monitor the status of all drones in real time and there must be mechanisms in place to prevent collisions between the drones. Also, the network must be capable of providing high bandwidth and low latency, which is especially important in SAR operations (Chua, 2013). Furthermore, task assignment for large numbers of UAVs has a high combinatorial complexity. Centralized fleet control architectures tend to suffer from communication overhead and having the central decision maker be a single point of failure for the system, whereas decentralized architectures are sensitive to information discrepancies across UAVs, which may lead to conflicting decisions (Bertuccelli, et al., 2009).

Missions for a drone fleet can be either preplanned or real-time, with the fleet moving in a formation or separately. Preplanned missions have the benefit of working even with poor connectivity, whereas real-time missions may be interrupted or run into risk of collisions when messages don't reach their endpoints. When the fleet is moving in formations or organized swarms, it is possible to have one or more UAVs flying on a higher altitude to act as coordinators for the operation (Rosalie, et al., 2016).

There is also some research into completely autonomous UAVs (Zhu, et al., 2017), (Ahmad, 2016). For example, there is a search method, in which the target area is split into cells and UAVs mark visited cells to

be avoided by their peers, so that the search can be carried out without a complete preset flight plan. This method was inspired by insects that use pheromones to guide their peers (Gaudiano, et al., 2003). There's also another approach called gradient optimization, where an uncertainty map is generated for the search area by surrounding points of interest with gradually decreasing uncertainty (Zhang, et al., 2017). The UAVs then autonomously search the area, trying to cover most of the uncertain territory, while avoiding collisions.

2.8 Classification

The European Association of Unmanned Vehicles Systems (EUROUVS) has classified UAVs into categories of micro/mini UAVs, tactical UAVs, strategic UAVs and special task UAVs. Micro UAVs have a maximum takeoff weight of 0.1kg and they come with a flight time of less than an hour. Although they are meant for scouting and indoor surveillance, their limited carrying capacity and endurance hinders their usability for search and rescue use cases. Mini UAVs on the other hand have a maximum takeoff weight of under 30kg and flight times up to 2 hours, which leaves room for heavier payloads and enables the coverage of much larger areas. The use cases for these UAVs include film industries, agriculture, communications relay and electronic warfare. (Bento, 2008)

The rest of the defined categories include drones that have a maximum takeoff weight between 150 and 12500kg and flight times from 2 to over 48 hours. Although some of these drones are mechanically better suited for search and rescue, they are also far more regulated and significantly more expensive to acquire and to maintain. Because of these issues, the heavier UAV categories are out of scope for this thesis.

2.9 Legislation

Regulations and rules regarding drone use differ from country to country. The primary things to consider are the weight of the drone, flight location and altitude and whether a license or certificates are required. Drone payloads should also be considered. Recording devices like cameras may have privacy concerns and the use of dropping systems may be restricted. Although the regulatory framework is fragmented currently, there are efforts from European Aviation Safety Agency (EASA), International Civil Aviation Organization (ICAO), Joint Authorities for the Rulemaking of Unmanned Systems (JARUS) and Federal Aviation Administration (FAA) to build a shared international framework of regulations for technical and operational requirements for drone usage. (EASA, 2017)

Finland

In Finland, drones have a maximum takeoff weight of 25kg, unless a special permission has been granted, which would require adhering to much stricter aviation law. Unlike many other countries, UAV pilots do not need any license or certificate of airworthiness. The drones can also be piloted beyond visual line of sight if the airspace has been reserved for the purpose. The maximum flight altitude is 150 meters, which is slightly more than the 120 meters or less that is set in many countries. Also, there are restrictions on flying above crowds of people. In Finland, like in most countries, UAVs must give way to manned aircraft. The drone doesn't have to be registered, but it has to carry a label that contains the name and contact info of the owner. (Joint Authorities for Rulemaking on Unmanned Systems, 2016)

2.10 Unmanned Aircraft System Traffic Management

Unmanned Aircraft System (UAS) refers to a system including UAVs, a ground-based controller and communications between them. NASA is developing a UAS Traffic Management (UTM) system to ensure safe drone operations. The system includes features such as airspace corridors, geofencing, weather avoidance, congestion management, terrain avoidance, re-routing. The end goal of the system is to make the system not require any human operators to continuously monitor vehicles. (NASA, 2017)

The UTM concept has also been worked on by Skyguide, who demonstrated registraton, identification, geofencing, flight planning, flight approval and dynamic airspace awareness in an event on 14.09.2017. Some of these capabilities are expected to be operational in 2019, although the services for providing them are ready for immediate deployment. (McNabb, 2017)

In addition to NASA and Skyguide, there are several other members collaborating under Global UTM Association to define the architecture for UAS traffic management. It is noted that emergency services will benefit from UTM because of the features to detect drones, prevent collisions, and to create drone-free zones for emergency procedures. (Global UTM Association, 2017)

For the use case of this thesis, the development of UTM means that safety features such as collision avoidance and no-flight zones do not have to be researched comprehensively, as they will be covered in the future by the global UTM systems. However, until that happens, the GCS or the UAVs themselves should implement basic collision prevention and no-flight zones.

2.11 Summary

This chapter introduced different types and classes of UAVs and described their properties. It was pointed out why multirotor vehicles under 25kg weight, controlled over LTE are currently the most viable option for the SAR use case. It was also discussed how the choice of power source might shift away from the common LiPo batteries towards hydrogen fuel cells in the near future.

SAR missions generally require speed, reliability and maneuverability. From the types and classes of UAVs discussed earlier, multicopters weighing under 25kg are the best match for the use case. Although fixed wing UAVs can reach significantly longer flight times, copters are more maneuverable and easier to take off and land. These qualities are important for search and rescue use cases as the vehicles must be launched quickly and during search missions it should be possible to stop them to take a closer look when items of interest are found.

3 Existing Practices for Search and Rescue

This chapter describes some of the common aspects of the SAR use case, focusing mainly on relevant search methods.

3.1 Overview

SAR operations are typically carried out with limited resources, which means that only a small area can be covered in the important first few hours of the search. When it comes to wilderness search and rescue for a missing person, every passing hour expands the area that needs to be searched by roughly 3 kilometers and it becomes less likely to find the person alive (Lin & Goodrich, 2010). If the missing person is not found within 51 hours of disappearance, the chances of finding them alive decreases significantly (Adams, et al., 2007). For Alzheimer's disease patients, this time limit is only 24 hours (Jurecka & Niedzielski, 2017). In urban disaster scenarios, victims' movements can be seemingly random, and thus very difficult to predict (Sánchez-García, et al., 2016).

3.2 Traditional Search Methods

The traditional method of searching for missing people is to send a team of people on foot to survey the area. This method still has some advantages over aerial search with UAVs. Most notably, the people will be able to see and hear their nearby surroundings much more accurately compared to a UAV. Furthermore, search dogs can be brought along to trace the people using their scent. However, the major drawback with this method is that despite being accurate, it is very slow to cover area, especially if the terrain is difficult to traverse. Also, it requires much more effort and dedicated personnel to carry out. (Perkins, et al., 2003)

Performing the search operation with UAVs instead is a way to trade some accuracy for a considerable increase in speed and area coverage.

3.3 Existing Search Methods for a Single Drone

The objective of a search is to find as many targets as fast as possible. There is a tradeoff between speed, detection, coverage and cost.

In many search situations, the main goal is to maximize the probability of finding the target. However, when it comes to SAR, it is more sensible to have a goal of finding the target alive. This means that areas that are more hazardous should be given higher priority even if they are otherwise considered to have a lower probability of containing the missing person. (Frost & Stone, 2001)

Uniform Full Coverage

In a simple case, there is no information about the location of the lost person, which means that the entire target area must be searched systematically. This might happen if the search area is uniform, for example, a region on the sea, or a patch of homogenous forest or plains.

Because multicopters must slow down and expend more energy during turning motions, it is best to minimize turns during search (Li, et al., 2011). This means that the most efficient method for fully searching

an area is by using the 'lawnmower'-pattern, in which the drone systematically scans the entire area going back and forth from one side to the other. For optimal performance, the pattern should be set up so that the back and forth movement goes along the longest side of the search area, which will minimize the amount of turns in the flight path (Maza & Ollero, 2004). This pattern is shown in Figure 13 below.

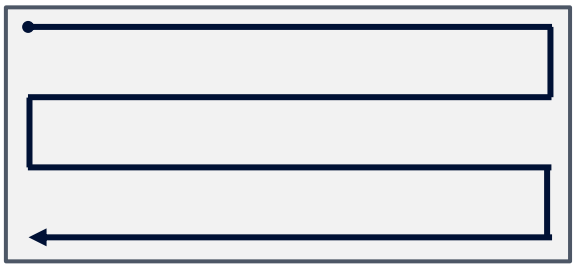

Figure 13: Lawnmower search pattern

The lawnmower pattern can be applied to any convex region using trapezoidal decomposition, where the area is subdivided into trapezoidal cells (Choset, 2001). Each cell can be then searched in sequence with the lawnmower pattern. This pattern is also commonly used for mapping missions, where pictures must be acquired with roughly even spacing from a large area. If the area contains obstacles that must be avoided, Boustrophedon decomposition can be used to take the obstacles into account during route planning (Leitner, 2009) (Choset, 2000). This method improves on the trapezoidal decomposition method by minimizing the number of lengthwise movements and thus covering the area faster. Both decomposition methods are shown in Figure 14 below. From the example figure, it can be seen that the trapezoidal decomposition yields flight paths with a total of 15 lengthwise movements, while the paths created using boustrophedon decomposition have only 13.

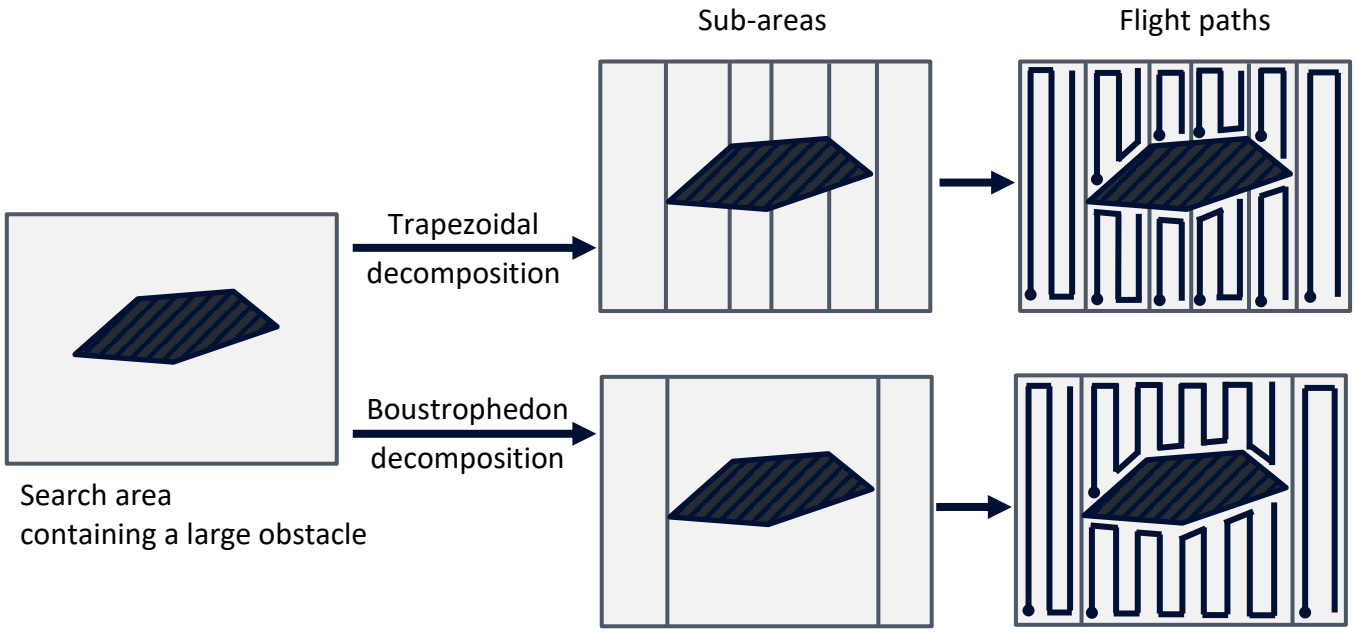

Figure 14: Area with an obstacle split into cells using trapezoidal deomposition and boutrophedon decomposition

An alternative to the lawnmower pattern search is Informative Path Planning (IPP) method (Popovic, et al., 2016). In this approach, the drone starts from the center of the target area at maximum feasible search altitude. The drone then starts descending and covering the area, following a dynamically generated flight trajectory that is updated throughout the search operation. The benefit of this approach is that it initially covers a large area with low accuracy, which will make it possible to detect any visually distinct features quickly. The drone will also keep descending to get higher quality footage. This method was designed for precision agriculture to detect weeds. However, it seems feasible for the search and rescue use case. A person's visibility may vary greatly depending on the type of clothing they are wearing so it would make sense to start the search from high altitudes in case the person happens to be wearing brightly colored and easily distinguishable clothes. In case nothing or nobody is found, then the drone can be brought lower for more precise search. Whereas a regular lawnmower pattern search will systematically scan the entire area with uniform precision, the IPP method starts with low precision and high coverage and moves towards high precision over time, making it a much more flexible and scalable method. In particular, if the time budget does not allow for the full coverage of the area, then the lawnmower pattern will leave portions of the area uncovered, whereas the IPP planning method will likely have at least low-quality footage of the entire area. Figure 15 below shows a comparison scenario where the IPP method and the lawnmower pattern have been used for the same amount of time to detect weeds from a given area.

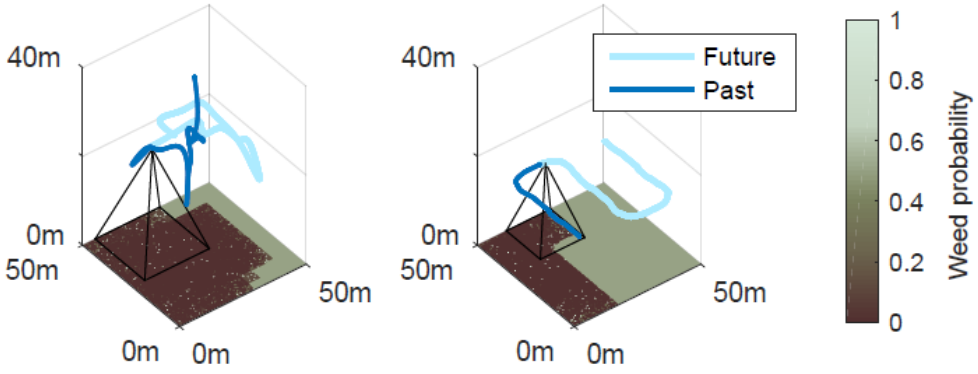

Figure 15: Informative path planning (left) compared to standard lawnmower pattern (right) (Popovic, et al., 2016)

Prioritized Coverage

In a real-life SAR situation, there is usually some prior information available that can be utilized to better perform the search operation. In the simplest scenario, the last known location of the missing person can be chosen as an initial planning point and a point of interest point for the search. In addition to this, the search area might contain known landmarks, hazards or other distinguished locations that should be considered important to search. In this case, the search scenario would involve multiple points of interest that should all be investigated in some order of priority. However, in the best-case scenario, there is detailed knowledge of the terrain, the profile of the missing person and details of past search operations in the area. This may make it possible to split the entire area into a grid composed of small cells, each labeled with an estimated probability of containing the person.

In the case of a single point of interest, a viable search method would be the expanding square pattern, in which the UAV is sent directly to the point of interest and then set to search outwards in a spiral pattern (Wollan, 2004). Because the drone has to slow down due to the frequent turns, this method covers area slower than the lawnmower pattern discussed earlier. However, this is acceptable because the highest priority areas are covered first. If the person is particularly difficult to detect, for example because of heavy vegetation or other obstacles, a sector search can be performed instead. This is performed by flying over the location repeatedly from different angles. This covers the area more accurately, but is also even slower to perform than the expanding square search since the same area is flown over multiple times. Nevertheless, it can be the best approach if the person's location can be narrowed down with sufficient precision. Both patterns are shown in Figure 16 below.

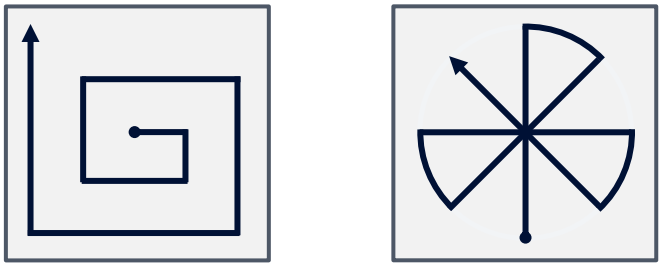

Figure 16: Expanding square (left) and sector search (right) patterns for investigating a single point of interest.

In cases, where there is more prior information available to assign probabilities to areas on the map, it is more beneficial to prioritize the most likely locations first before searching elsewhere. Although it is far from a trivial task, these probabilities can be derived from the terrain, vegetation, weather and known information about the lost person (Lin, 2009). One method of utilizing this prior information is by marking several of the highest priority locations on the map as points of interest. The full search area can then be split into sub-areas around the defined points and UAVs can be assigned to search these areas in order of priority. One method of performing this split is to use Voronoi tessellation, in which the area is split into polygons around the predefined points, so that borders between polygons are at an equal distance from the points. Splitting an area with four points of interest using Voronoi tessellation is illustrated in Figure 17 below.

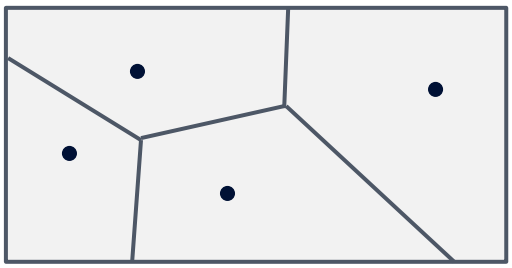

Figure 17: An area split into sub-areas using Voronoi Tessellation

Ideally, the probabilities of the missing person's location should be updated as the search progresses based on environmental conditions, search results and the profile of the missing person (Adams, et al., 2007). This information should then be used to update the flight plans dynamically.

From usage point of view, it is beneficial to be able to give high level orders to drones, such as ordering a drone to search a certain area for a set amount of time. If the user focuses on high-level strategic planning, leaving the details to the drones, they can dedicate more time to observing video feeds and communicating with other team members (Lin, 2009).

3.4 Expanding to multiple drones

The search methods presented earlier have all been designed for a single UAV. However, there are two methods to scale up the previously presented search methods for a multi-UAV scenario. The first method is to organize the UAV fleet into a search formation, which can then be considered as a single unit with a considerably larger field of view. When a search pattern is generated for such formation, the lines of the path will be much further apart and the area is covered faster. The other method is to simply split the search area (e.g., by using Voronoi tessellation on points of interest) into separate sub-areas, and allocating a single drone to each area. This second method is not as easy to accomplish as the formation search, but it has minimal flight time, and thus has better performance (Kou, et al., 2017). Both of these methods are shown in Figure 18 below. These methods can also be combined to have individual drones and small drone formations, each working on their own designated areas. Finding out the appropriate approach for different scenarios is something that needs to be determined on a case by case basis.

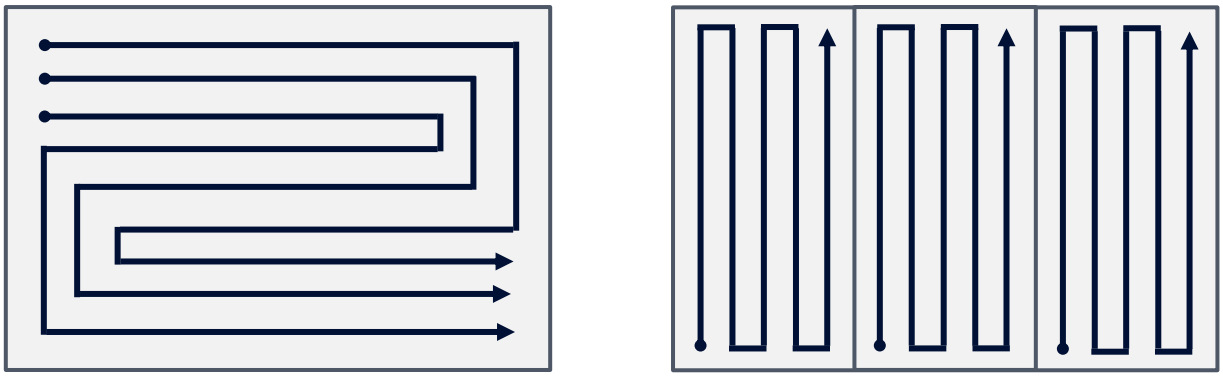

Figure 18: Lawnmower search pattern using a fleet formation (left) and individual drones (right)

3.5 Summary

This chapter presented known methods and requirements for performing search of missing people. It was noted that UAVs can cover area much faster than human search parties on foot and that UAVs can reach places that may be impossible to reach with manned vehicles, due to their lack of maneuverability.

The lawnmower pattern search was introduced as a solution for fully covering large areas when there was no prior information available to help prioritize the search locations. Trapezoidal and boustrophedon decomposition methods were discussed as ways to split a non-rectangular area into sub-areas that can be searched by directly applying the lawnmower pattern. For uniform coverage without obstacles, the informative path planning method was introduced as a more efficient area coverage method.

The expanding square and sector search patterns were introduced for the scenario where a location is known beforehand. And for the scenario where multiple points of interest were known, Voronoi tessellation was

introduced as an algorithm to split the search area according to the points into sub-areas that could be searched in sequence or simultaneously by one or more drones.

It was noted that all the introduced search patterns were designed for a single UAV, but that they can be easily expanded to a fleet scenario either by grouping up the UAVs into a unified formation, or by dividing the search area into sub-areas to be searched by individual UAVs or small formations.

In the use case of this thesis, the introduced methods can be used by the GCS operator to design flight paths for the UAVs. Ideally, the GCS would also include algorithms to generate flight paths according to these patterns to any given area, simplifying and speeding up the workflow for the user.

4 Existing Drone Control Systems

There are several existing GCS applications available for controlling drones. This chapter evaluates and compares some of them.

4.1 Overview

A GCS for controlling any remote unmanned vehicles has two main functions. It must provide the user with control over the vehicles and it must grant easy and effective access to the data collected by them (Arnold, 2016).

There exist many open source flight control applications, which use the popular MAVLink micro air vehicle communications protocol to communicate with compatible autopilots, such as PX4 and APM. On the other hand, drone manufacturers often provide their own proprietary GCS applications for controlling the drones they make. There are also some applications that are meant to work with multiple different types of vehicles. The following comparison of existing ground stations starts with the available open source applications, followed by proprietary and universal GCS applications.

4.2 Open Source Ground Stations

Open source ground stations have the benefit of being modifiable and extendable by the user. However, they are often considered to be more difficult to use compared to other options, because of the amount of configuration needed or because the user experience has not been a major priority in the project.

Mission Planner

Mission Planner is an open source GCS application for ArduPilot-based vehicles. It comes with features for vehicle setup and configuration, route planning, control and log analysis (Oborne, 2016). Because the software has been developed as Windows Forms application, it is only available for the Windows operating system. Although it is technically possible to run the software on Linux or Mac using Mono, it is not recommended for performance reasons.

The software has features for generating simple area survey routes, and the user can give direct waypoints in real time or plan a route directly by creating waypoints on a map. However, it is primarily designed for controlling a single drone, which means that the user cannot plan multiple routes simultaneously or give simultaneous controls to multiple drones. One exception to this is the experimental drone swarming feature, which enables the user to connect multiple drones and to create a formation around a designated leader drone.

Mission Planner is very popular and has been used in UAV related research (Sullivan, 2016). Also, there have been projects where it has been used as a base to build a new GCS application on top of it (Mallesh, et al., 2015). It has been noted that the software lacks 3D planning and does not support splitting long flights into multiple separate missions, which can be important when planning routes that cannot be flown without replacing the battery (Gandor, et al., 2015). The user interface of Mission Planner is shown in Figure 19 below.

Figure 19: Screenshot of Mission Planner user interface (Oborne, 2016)

QGroundControl

QGroundControl is a multiplatform ground control station developed with Qt framework, which allows it to be deployed to Windows, Mac, Linux, Android and iOS. It provides control and setup functionality for PX4 and APM vehicles. It is also the GCS of choice for the rather large open source collaborative Dronecode project (Dronecode Project, 2017).

In contrast to Mission Planner, QGroundControl is considered less intuitive and more difficult to use for less experienced users (Gandor, et al., 2015). It is shown in Figure 20 below.

Figure 20: QGroundControl (Gagne, 2017)

APM Planner 2

APM Planner aims to combine the easy user interface of Mission Planner with the multiplatform support of QGroundControl. It can be run on Windows, Mac and Linux. However, it is missing some of the more advanced features of Mission Planner, such as access to the full parameter list and some of the mission commands, and it has a smaller user base (ArduPilot Dev Team, 2016). A screenshot of the user interface is shown in Figure 21 below.

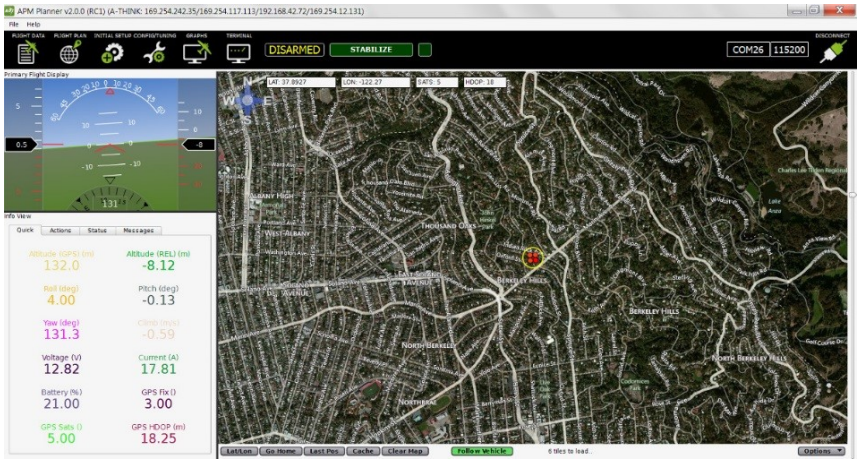

Figure 21: APM Planner 2.0 (ArduPilot Dev Team, 2016)

MAVProxy

MAVProxy is also a GCS for MAVLink based UAVs. It has been designed as a fully functional minimalist application that is both portable and extendable (Tridgell, et al., 2016).

As its name implies, MAVProxy can also be used as a proxy to forward traffic between ground stations and autopilots. For this reason, it is often used only to forward incoming MAVLink traffic between various endpoints, while some other system is used to perform as the actual GCS (Choi, et al., 2016), (Birnbaum, et al., 2015). There has also been drone-related research where a custom GCS was built on top of MAVProxy (Wang, et al., 2015).

The user interface of MAVProxy is shown in Figure 22 below. It consists of three separate windows: a map window, an output console, and a control console. The drone is controlled by using a command-line to input commands. The minimalist design makes MAVProxy light enough to be run on small netbooks without performance issues.

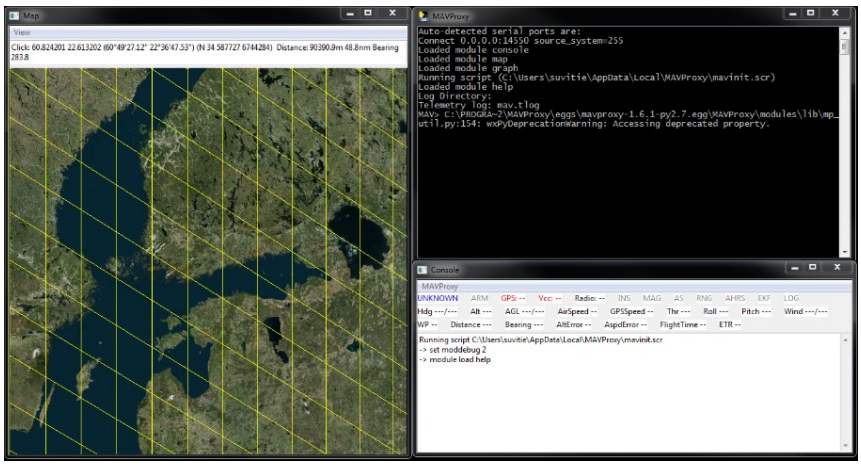

Figure 22: Screenshot of MAVProxy 1.6 (Tridgell, et al., 2016)

Droid Planner Tower

Tower is an Android GCS application for controlling ArduPilot UAVs. It is meant to be used with a tablet and it comes with the basic route planning and control features as well as a tool for performing area surveys. The software also supports displaying video streams from the drone and there is an option for offline maps. There's also a 'follow-me' feature which is used to make the UAV follow the tablet. The user interface of Tower is shown in Figure 23 below.

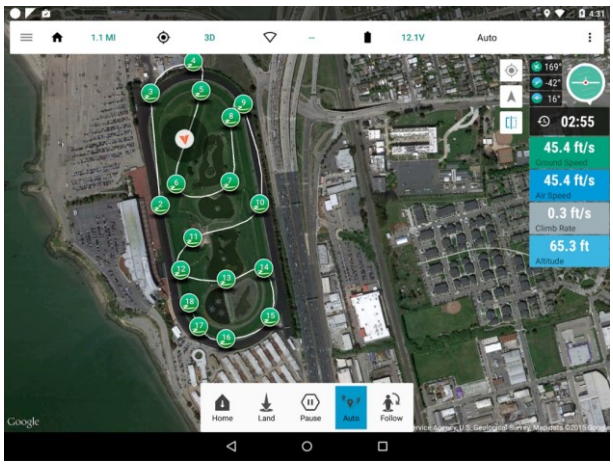

Figure 23: Tower (DroidPlanner Labs, 2017)

4.3 Proprietary Ground Stations

Proprietary ground stations are closed source applications designed to work with a specific set of vehicles. This means that the software itself typically cannot be modified or extended by the user and the user often does not have access to modify all the vehicle's settings and parameters. However, as a direct result of this lack of options, proprietary ground stations tend to be very easy to use. Most of the configuration and setup work has been done in advance by the developers and the user is not given the option to give commands that could cause disaster.

3DR Site Scan

Site Scan is a GCS application for iPad tablets running iOS and it is primarily meant for performing 2D and 3D mapping for industries. It comes with tools to generate flight routes for surveys to inspect areas or buildings from multiple angles. Because the user has to only define the areas and choose the mission type, the flight planning is faster than it would be if the user had to manually input the waypoints for the route. This GCS can be used with either DJI or 3DR drones. The interface is shown in Figure 24 below.

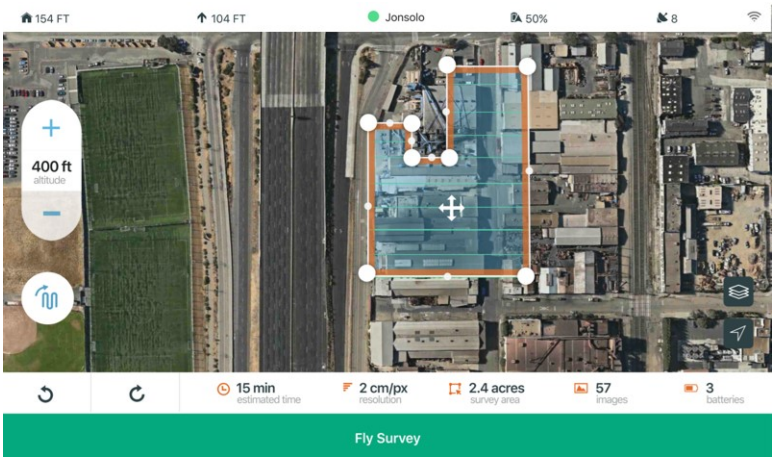

Figure 24: Site Scan (3D Robotics, Inc, 2017)

DJI GS Pro

GS Pro is a proprietary iPad application for controlling UAVs developed by DJI. It also comes with features to perform area surveys and structure surveys. These are done by having the user specify the target area on a map and then specifying mission-wide parameters such as flight speed, altitude and photo overlap ratio.

They also highlight a 'Virtual Fence' safety feature, which allows the user to specify the UAVs altitude and speed within an area. The interface of the software is displayed in Figure 25 below.

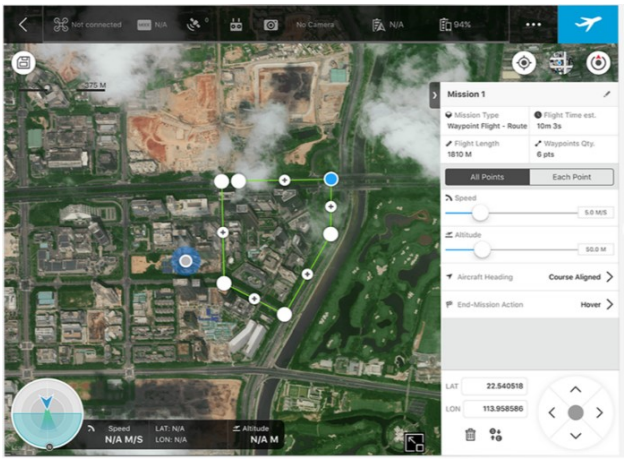

Figure 25: GS Pro (DJI, 2017)

Litchi

Litchi is another application for controlling DJI manufactured drones. While also being available for iOS devices, it differs from GS Pro by being usable with a web browser and for android devices. It includes features to plan basic waypoint missions, to take panorama shots and to follow the user. Screenshot of this application is shown in Figure 26 below. (VC Technology Ltd, 2017)

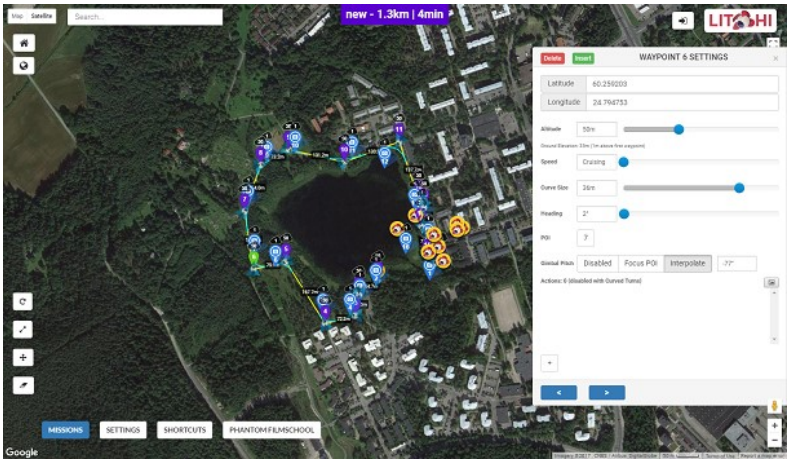

Figure 26: Screenshot of Litchi web browser

4.4 Universal Ground Stations

Universal ground stations are designed to support multiple platforms from the same interface. They are often developed with a modular design that makes it possible to integrate new platforms with minimal effort. In comparison to the GCS applications discussed earlier, universal GCS applications require the user to spend some time configuring the interface or mission plan for the specific devices they are going to use.

UgCS

UgCS is a proprietary GCS application developed with the Unity engine, available for Windows, Mac and Linux (SPH Engineering, 2017). It differs from previously discussed software by simultaneously supporting autopilots from multiple different manufacturers. This is accomplished with vehicle specific modules that translate the commands according to the target autopilot (Brass, 2015). This makes it possible to carry out missions with a UAV fleet that is made up from vehicles, which are not all developed by the same manufacturer. However, this feature does bring an extra step to the workflow, as the user has to manually specify the type of vehicle and payload for flight routes.

Instead of having a two-dimensional map view like most other GCS applications, UgCS features a three-dimensional map view. This helps in planning flight paths especially for areas that have steep hills or large buildings. The usefulness of this feature depends on the availability of building models and the accuracy of the terrain elevation data for the flight location. Also, navigating a 3D map is a more complex task than navigating a 2D map, because the user has to manage the viewing angle in addition to the map location and zoom level. This may take the user some time to get used to.

UgCS supports connecting to multiple UAVs simultaneously. The software also enables the user to create virtual simulated drones, which can be used for practice. The user can select any connected drone, upload a planned flight route and issue commands. However, the software only allows the user to select and control a single drone at a time.

The developers do provide another application, the ‘Drone Dance Controller’, to enable the simultaneous control of multiple drones. This application was designed to perform choreographed outdoor drone shows

using MAVLink based vehicles. It is meant to be used together with UgCS and it can manage up to 20 simultaneous drones (SPH Engineering, 2017). Both UgCS and this application are shown in Figure 27 below.

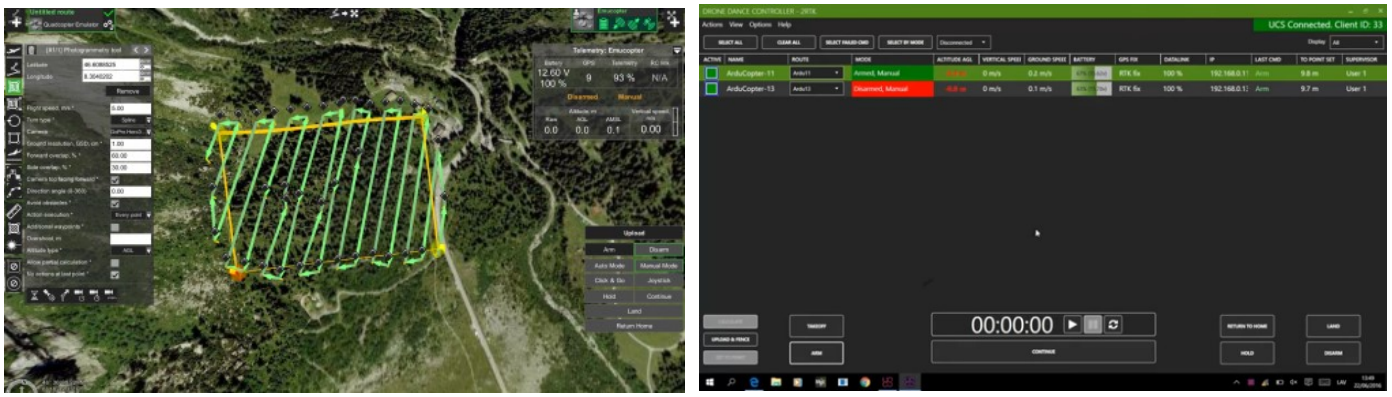

Figure 27: UgCS (left) and DroneDanceController (right) (SPH Engineering, 2017)

AMFIS

AMFIS is a generic surveillance system designed to work with UAVs, unmanned ground vehicles and stationary sensors (Bürkle, et al., 2010). As can be seen in Figure 28 below, the user interface of AMFIS has been designed for three workstations. It is meant to be used by two operators, who both work on their own workstation. The first operator works on the data, marking important information and relaying it forward, while the second operator controls the vehicles and sensors directly. The middle display is dedicated to situation awareness and is meant to be used by both operators. The UI is designed to be fully adaptable, where various elements can be activated or deactivated to customize the view for the scenario and devices being used.

AMFIS comes with a flight path planning tool, which lets the user define a polygonal shape on a map, select one or more connected UAVs and enter flight parameters. The software then calculates paths to cover the area and perform a mapping survey. The software also includes a virtual drone simulator, which is meant to be used for training operators.

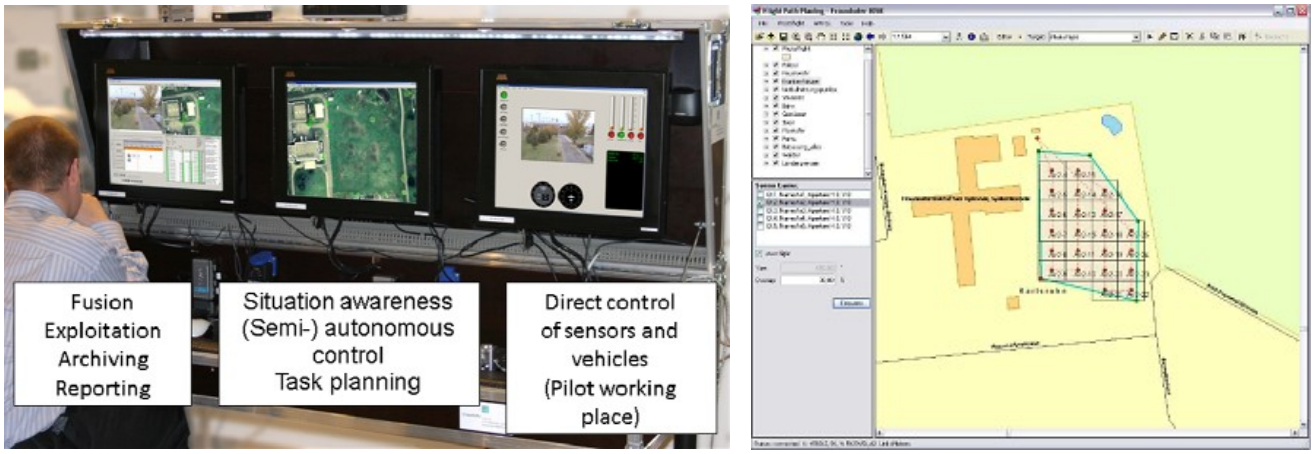

Figure 28: AMFIS (left) and its Flight path planning screen (right) (Bürkle, et al., 2010)

4.5 Summary

There are many GCS applications available and after taking a look at several of them, it can be determined that the most basic features among GCS software are the following:

- Map that displays UAV location and routes
- Waypoint-based route planning on the map
- Real-time UAV status display
- UAV mission controls
- Real-time guided waypoint controls

These features were present in all ground stations researched. If any of these features were missing, the software would be unusable. Additional features that brought extra value to the GCS were the following:

- Automatic route generation to given area with adjustable parameters
- UAV calibration and setup
- Connecting to multiple UAVs simultaneously
- Planning multiple routes
- Controlling multiple UAVs simultaneously
- Swarm formation
- UAV video stream display
- Virtual drone simulation
- GCS available on multiple operating systems
- Support for multiple autopilots
- 3D map view
- Offline maps
- Support for multiple displays
- Support for multiple operators
- Manual flight using joystick

Most of the currently available control software for drones seem to be mainly focused on the use case of flying a single drone to perform aerial photography or mapping.

5 Specification of a Search and Rescue Drone System

This chapter describes the requirements for a fleet control software, followed with the requirements for drone configuration.

5.1 Requirements for Fleet Control Software

Controlling a fleet of drones requires the use of a dedicated GCS, which provides the capability for status monitoring, route planning and control for multiple drones.

At the end of Chapter 4, various GCS features were listed as being either a part of the basic functionality of a GCS or a feature that simply added value to the software. Those features have been prioritized for the fleet search and rescue use case in Table 3 below.

Table 3: GCS features prioritized for the fleet search and rescue use case

Must have	Should have	Nice to have
Map that displays UAV location and routes	Offline maps	UAV calibration and setup
Waypoint-based route planning	Automatic route generation	UAV video stream display
Real-time UAV status display	Virtual drone simulation	Multiplatform GCS
UAV mission controls		Support for multiple autopilots
Real-time guided waypoint control		3D map view
Simultaneous connection to multiple UAVs		Support for multiple displays
Simultaneous control of multiple UAVs		Support for multiple operators
Planning of multiple routes -OR- swarm formation support		Manual flight using joystick

Few of the features are considered essential for the software and they are in the ‘Must have’ column in Table 3 above. The map display is a central component of any GCS because it shows the environment, the current location of vehicles, as well as the planned routes. Waypoint-based route planning is the most basic form of planning. Even though the routes could also be automatically generated, it is still important to keep it possible for the user to adjust the details of the route when necessary. The real-time status display is critical in being able to detect and prevent problems that might cause the UAV to crash. Mission controls are needed for running preplanned routes and real-time guided waypoint controls are necessary when something of interest is spotted and needs to be investigated without delay. Also, being able to connect to and control multiple UAVs simultaneously are essential features for the fleet search and rescue use case. Finally, it is necessary to either have the ability to plan multiple routes or to be able to control the UAVs as a single formation. These requirements come from the two approaches to expand search patterns to multiple UAVs, which were presented in chapter 3.4.

In addition to the essential features, there were three high-priority features that would add much value if implemented, but would not make the software completely unusable if they were missing. Offline maps would ensure that the software remains usable when there is no internet connection available, which may well happen in a disaster scenario. Another high-priority feature is automatic route generation, because it speeds up the route planning phase considerably, which is important for time-critical search missions. It is

not considered a ‘must have’-feature because it is still possible to successfully carry out a search mission with routes that are manually planned by the operator. Simulated virtual drones are useful for training users of the software, but they are also very valuable for testing while the GCS is being developed. For this reason, it would be highly beneficial to implement them at least to some degree.

The rest were considered as ‘nice to have’-features, which would make the software more convenient to use, but would not have a major impact if they were left unimplemented. The calibration and setup of UAVs is not considered important for the GCS because it is not something that needs to be done all the time and because there already exists some software that can be used to perform these functions for every autopilot. Video stream display is critically important for the use case, but because it can be done outside the GCS software with existing applications, it is only considered a convenience feature for a GCS. Supporting multiple operating systems for the GCS or multiple autopilots would make the application more flexible, but as long as the system supports one operating system and one type of autopilot, it will be fully functional. Additionally, it is worth noting that supporting multiple platforms means that the overall development of the software may be slowed down by platform specific issues. 3D map view is very convenient for previewing routes, but it adds complexity to map navigation and after trying it a while, it was evident that the terrain elevation does not vary enough outside of very mountainous regions for any practical benefit. Support for multiple displays means that the application UI can be adapted to be spread on more than one screen. Ideally, the software should be fully usable with just one screen, but being able to adapt to more screens would be a convenience. Support for multiple operators would mean that the same software could be used by multiple users cooperatively, which may be useful in a scenario with a large fleet. However, for the sake of simplicity it is not considered very high priority. Manual flight using a joystick is also not considered a high priority feature because a dedicated radio controller can be used to do it. The radio can also be used as a backup control mechanism if there is any issue with the GCS.

Overview

In addition to route planning and drone controls, the control station software must also display drone status information, which allows the user to estimate remaining flight time and to react to any dangerous situations.

Route planning is commonly implemented so that the user can place consecutive waypoint locations on a map. A simple waypoint consists of only values for latitude, longitude and altitude. A route may also contain certain commands in addition to waypoints. For example, the route can be set to begin with a Takeoff command and to end with a Land command, in which case the user does not have to manually issue these commands.

Centralized vs Decentralized Decision Making

Centralized decision making of a drone fleet is usually impractical due to communication limits, robustness issues and scalability (Mirzaei, et al., 2011). These concerns need to be considered when developing a ground station that acts as a centralized control point for a drone fleet.

The communication limits consist of coverage and bandwidth. With LTE connections using a portable ultra-compact network, the coverage can be extended up to 75km (Skinner, 2016). When compared to traditional radio connections, a well-planned and installed LTE network has significantly greater range and it does not suffer from attenuation caused by flying behind large obstacles beyond visual line of sight (Sundqvist, 2015).

A sudden loss of the ground station during mission, either due to hardware or connection failure, is a very big problem. A simple solution to this issue is to use pre-planned missions, which the drone can follow to the end even when connections are lost. In this scenario, the drone should store any payload output information locally, so that it can be extracted and used after the drones return from the mission. The autopilot comes with built-in features for failsafe behaviors, such as having the drone return to launch when battery is empty or the connections are lost.

Scalability becomes relevant when a significant number of drones are used as a fleet to perform missions. In that case, the drones should be sorted or categorized by relevance, and bandwidth should be allocated accordingly. The bandwidth consumption of a drone can be adjusted by configuring the telemetry rates and payload stream qualities, which has to be done dynamically on runtime.

5.2 Requirements for Drone Configuration

Several LTE-equipped drones were built to be used in swarm flights. This chapter describes how they were built and set up.

Drone Components and Architecture

The basic components required by any drone are: Frame, Propellers, Motors, Electronic speed controllers, Flight controller and Battery. Carbon fiber frames and propellers were chosen for this project for their durability and light weight.

The first experimental flights were done using custom built quadcopters with an open source autopilot connected to an onboard computing board and an LTE modem.

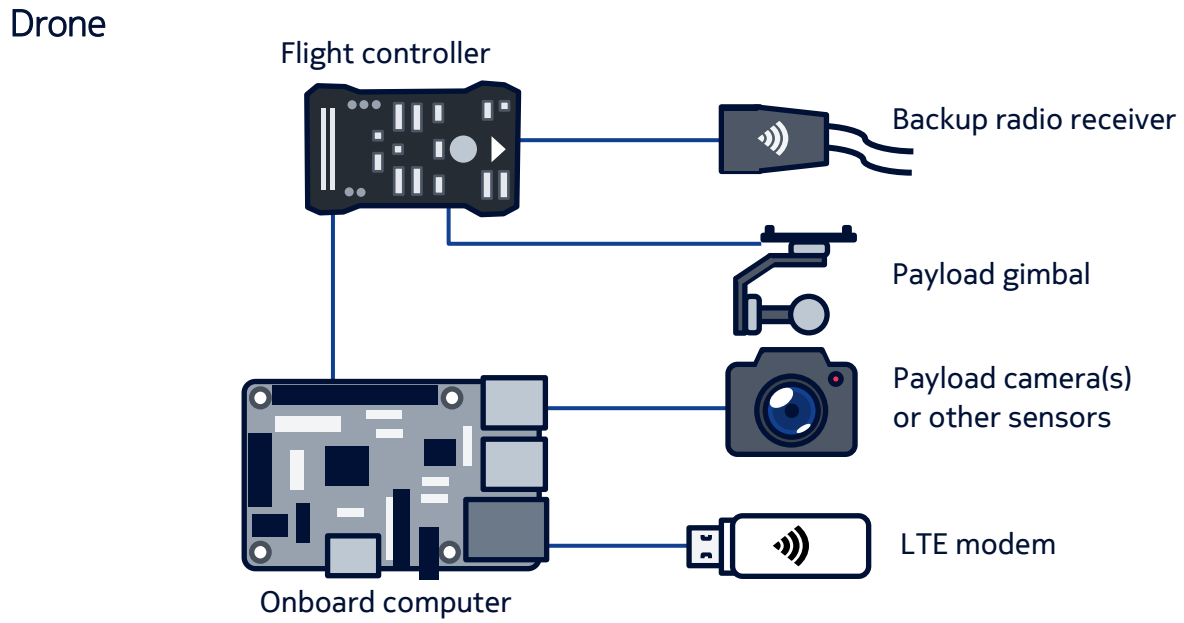

Figure 29: Main components of a drone

The main components of an LTE-connected drone are depicted in Figure 29. The flight controller controls the motors and handles all low-level autopilot logic. During this thesis, the chosen flight controller was a Pixhawk running open source ArduCopter firmware. The flight controller was connected to an onboard computer for control messages and to a radio receiver, that could be used as backup control method, which enables taking over the drone with a traditional radio controller in case the LTE connection is for some reason lost. Also, the payload gimbal was directly connected to the flight controller.

The onboard computer carries all the custom software that is required to make the system work. At the very least, it must connect the flight controller to the ground station via LTE and provide access to the payload data such as video feeds. Ideally, all delay-sensitive logic will be stored in the onboard computer to ensure responsiveness. The drones were controlled over LTE using the custom-made fleet control software was developed as part of this thesis.

One of the quadcopters built for test flights is shown in Figure 30 below. On top of the drone, there is the external GPS module. Below it, on top of the onboard computer board are the radio receiver and the LTE modem. Under the onboard computer, there's the flight controller and a 6-Cell LiPo battery is fixed to the bottom of the frame.

Figure 30: A quadcopter used in a test flight

Drone Calibration

Before first use, the drone has to be calibrated in order to prevent it from crashing or flying away. Drones carrying a Pixhawk autopilot can be conveniently calibrated using Mission Planner (Oborne, 2016), which comes with built-in features for calibrating the compass, accelerometers, battery levels and PIDs. From practical field tests, it was discovered that sometimes calibrations need to be redone if the hardware configuration has been changed or if the drone has not been used in a long while. It turned out that the compass required calibration the most often.

Drone Pre-Flight and Test Flight

In order to have a successful and safe flight, there are a few things to inspect in the drone beforehand. First, it should be ensured that the battery is in good condition. In the case of LiPo batteries, this means that it must not be swollen. Then, the drone should be powered on and it should be made sure that it can connect to the ground station and to the backup radio. If the radio doesn't work, it might not be bound to the drone or the transceivers might be broken. If the ground station fails to connect, the drone should be inspected to make sure that it is configured to connect to the right address, and that the LTE modem has successfully connected to the correct network on both the drone and the ground station computer.

Once the drone connects to ground station, it should be made sure that the drone is in flying condition. The telemetry from the drone should indicate that the battery is full and that there are enough satellites for the GPS to function. Also, there should be no error messages sent from the autopilot.

The radio controllers can be configured to trigger different flight modes by using switches. Because pilots may have different preferences, the mode switches should be tested before flight to make sure that they have been set up correctly. This will prevent issues if the drone has to be taken into manual control during the flight for any reason.

The drone reports Extended Kalman Filter (EKF) values to indicate the accuracy of various sensors. These values should be checked both before the flight and during the first test flight to make sure there are no issues with the sensors. To do the test before flight, it was found to be good to pick the drone up and rotate it full circle along the yaw axis while paying attention to the compass EKF values. If the compass is not properly calibrated, or if the drone frame has issues arising from magnetic components or payloads, the compass EKF value will jump high when the drone is rotated. This issue can be fixed by recalibrating the compass, or if that fails, by inspecting drone components. During drone setup on a few occasions, the location of the drone seemed to sometimes affect the EKF readings. It may be possible for the environment to cause compass issues, so it is a good idea to make sure the drone is not taking off from above an underground power line, for example.

If the drone has not been flown recently, it is always a good idea to perform a quick test flight. The drone should be flown manually to a low altitude and rotated around the yaw axis, while observing that the reported EKF values remain low, the drone moves as expected, and it doesn’t produce odd sounds.

6 Prototype of a Search and Rescue Drone System

This chapter goes into detail about the work done to customize an existing open source GCS application for the fleet search and rescue use case. The chapter starts with a short overview of the overall system architecture, followed by describing the development methodology and the actual modifications that were built on top of the open source GCS. Finally, it presents the outcomes and learnings of field tests, which are used in the following chapter to help in the creation of the new GCS application.

6.1 Architecture of the System

A high-level overview of the communications between different system components is presented in Figure 31 below. The flight controller handles the basic behavior of the drone, while the ground control station is the primary means of mission planning and control. The radio controller is always present as a backup control method in case something happens to either the GCS or the LTE connection. A server can be included as an optional component to store the telemetry and payload data streamed by the drone. It can also be used to perform functions that are too intensive to be run on the drone's onboard computer, such as pattern recognition or route calculations for complex scenarios.

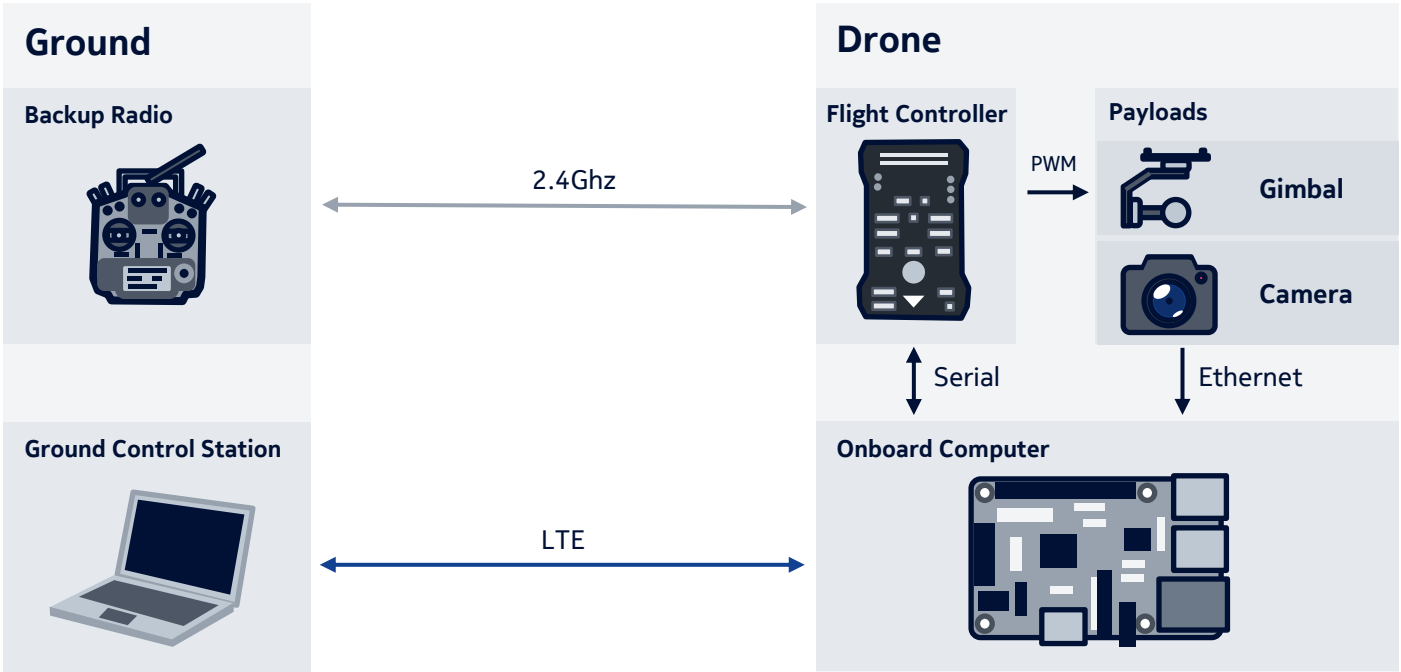

Figure 31: High-level overview of system communications

6.2 Methodology

Mission Planner was chosen as the base to build the prototype on. The reason for this choise was that it has many features, there are plenty of online resources available for it due to its popularity and it has been successfully used in earlier research to develop a new GCS on top of it. These features made Mission Planner seem the least risky option out of all the other available open source GCS applications.

As mentioned in chapter 4, Mission Planner is a Windows Forms application. This means that development is done with the C# language using the Visual Studio integrated development environment. Building the project produces an executable file, accompanied by any included Dynamic Link Libraries (DLL). During development, the software was also built and run in debug mode, which enables the use of break points to stop program execution at specified lines of code, and the inspection of the contents of variables.

The work on the modifications for Mission Planner started with studying the source code, following the existing program logic and making notes to gain an understanding of the architecture and design choices behind the software. Roadmaps were made and kept up to date for the required features and their priorities.

The work also required some reverse engineering by using the Wireshark packet analyzer to quickly discover what types of messages and parameters the software used in various situations. This was very helpful because the online documentation for the MAVLink communication protocol turned out to be rather inaccurate. On multiple occasions while working on this project, it was found that many of the message definitions were either deprecated or not implemented yet. Furthermore, some of the message parameters were silently ignored by the autopilot. Looking through the contents of packets that were sent from Mission Planner was the quickest way to discover messages and parameters that were guaranteed to work. Also, because of this issue, it was necessary to do field tests with small test drones often to make sure that the autopilot would interpret the commands correctly. One of these test drones is shown in Figure 32 below.

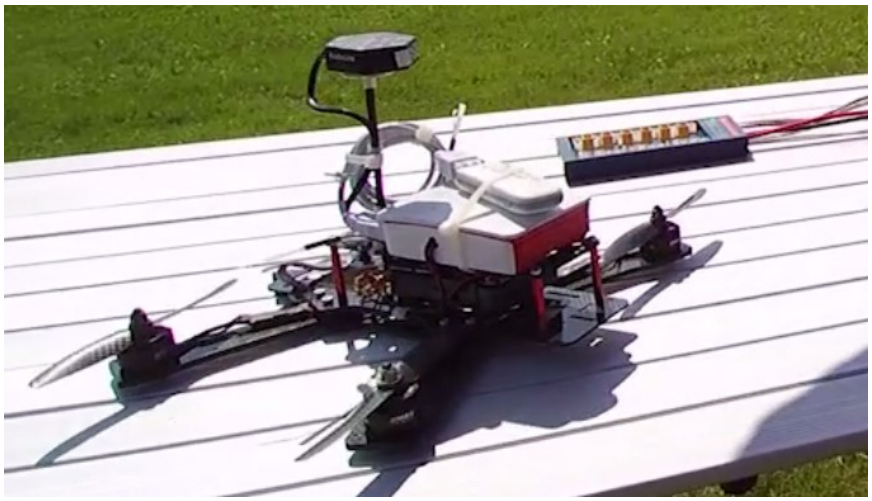

Figure 32: One of the small expendable test drones

6.3 Ports

Because the initial test drones were tedious to configure, enabling multiple drones to connect via User Datagram Protocol (UDP) through the same port was one of the first modifications to be done. This seemed to work fine initially, but it later turned out that the MAVLink packets sent by the drone were occasionally split into multiple UDP packets, which caused packet loss with this new modification. The reason being that the MAVLink parser simply receives byte arrays as input and pieces together new messages once all the bytes have been received. The issue with the single port implementation was that a fragmented MAVLink packet might be followed by a packet from a different drone, which would cause the parser to discard the

early bytes as an invalid message. Later, once the startup scripts for the drones themselves were improved sufficiently to make it convenient to define a unique port for each drone, the packet loss issue was resolved.

6.4 Swarming

The version of Mission Planner that was modified came with an experimental drone swarming feature, which seemed to be a good base for implementing formation-based drone flights. This feature worked by designating one drone as a leader and other connected drones as followers. The followers were periodically sent guided waypoints, that were calculated as offsets from the leader's location. The user could drag drone markers on a grid to define desired offsets and make up arbitrary formations. This interface is shown in Figure 33 below. On the grid, 'up' meant towards the facing direction of the leader drone.

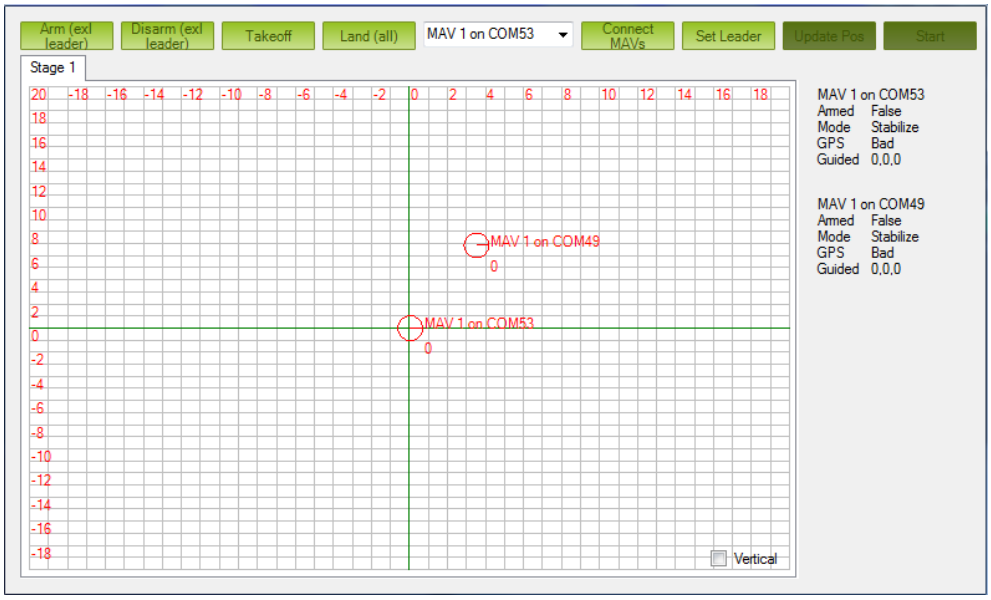

Figure 33: Mission Planner swarming feature (ArduPilot Dev Team, 2016)

This feature was expanded in the proof of concept project so that instead of having to manually specify the drone offsets, the user would instead input the planned flight altitude and the field of view angles of the payload cameras. These values would be used to automatically generate a formation that would keep the drones optimally spaced apart to both cover maximum area and to keep safe distances to each other. The generated formation starts out as a line, where the distance between the drones was calculated from horizontal camera angle, altitude, and required minimum overlap of camera footage. After this, the specified minimum safety distance between drones was checked. If the drones were too close to each other in the line formation, then every other drone was moved backwards so that the sideways spacing between drones remained the same and guaranteed full area coverage, while the added backwards spacing ensured safe distances between the drones. If the backwards spacing had to be added, then the straight-line formation turned into a zigzag formation. The formation control interface is shown in Figure 34 below.

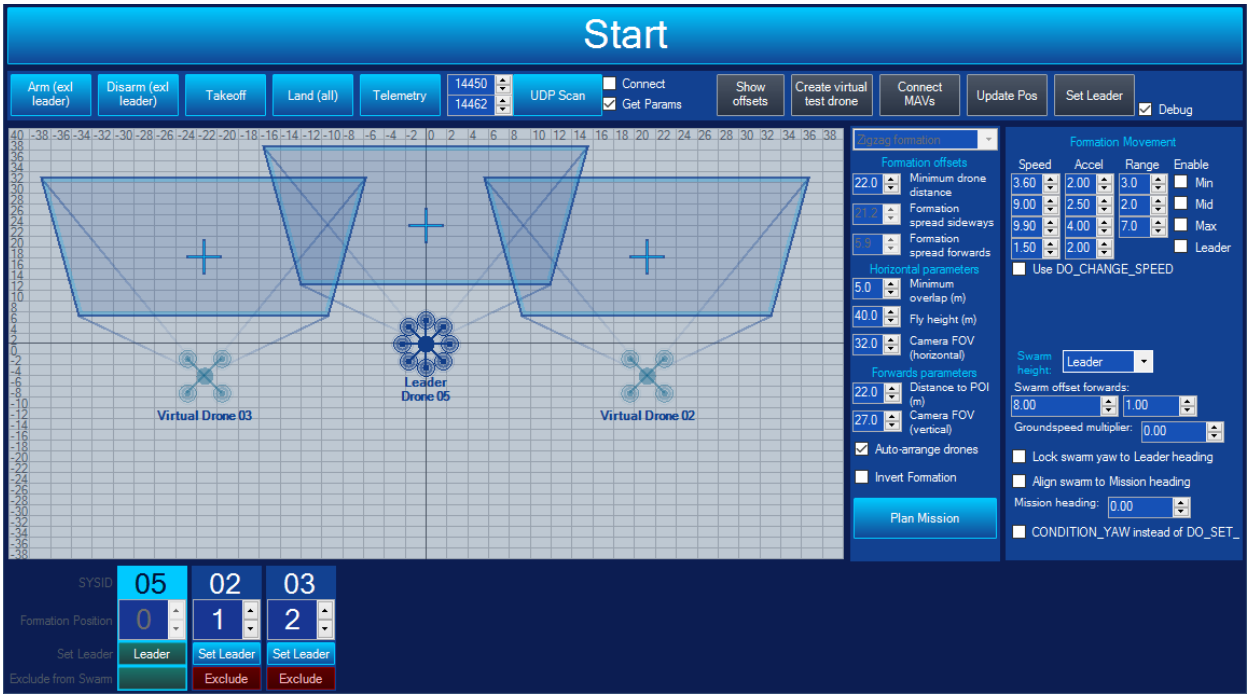

Figure 34: Formation controls built on top of the Mission Planner swarming feature

6.5 Controls

To enable the monitoring and control of multiple drones simultaneously, a new window was created to display the status of all connected drones. This window was created by taking the existing telemetry column and creating and instance for each connected drone. The telemetry columns were modified to include the following:

- ID of the drone
- Button for arming/disarming
- Indicator for the swarm position index
- Debug text display
- Button to start/stop the swarming feature for the leader drone
- Button to ignore the swarm for swarm members
- Buttons for takeoff, landing and return to launch
- Battery indicator bar

These changes can be seen in Figure 35 below. The swarm formation position was initially created as a numeric control that would enable the user to adjust the formation from this screen. However, this control was disabled when it turned out that the user should never switch the places of the drones during flight because there would be a very high risk of collision. Also, the artificial horizon of Mission Planner already displayed battery voltage in the lower left corner. However, this was only the total voltage and it was easy to miss, especially when there were multiple drones, so a new clearer indicator was built. The new battery indicator displayed total voltage, cell voltage, and the percentage remaining with a colored bar that went from green to yellow to red. This made it possible to find out the battery status with a glance.

Figure 35: Original Mission Planner telemetry column (left) extended to multiple drones (right)

While this new window made it much easier to keep track of drone status and to send commands to specific drones, it took a lot of space and didn't let the user send commands to all drones simultaneously. To fix these issues, another UI control element was created. In this new element, the information and controls of all connected drones were combined into a single column. This column included a table that displayed the most relevant information of each drone, color coded to draw attention to any abnormal values. The same table also had buttons for all of the commands for individual drones that were present in the earlier design. The battery indicator was made larger and it was set to display the battery status of the drone with lowest voltage, making it easier for the user to estimate when the drones should be brought back. Also, buttons for swarm-wide controls were implemented. These buttons enabled the user to send commands to all connected drones simultaneously, which simplified the workflow significantly. Also, swarm flight parameters, such as speed and altitude, could be modified directly in this new interface. Figure 36 below shows this new UI element both in early stages of development and towards the end.

Figure 36: Swarm controls during early development (left) and towards the end of the project (right)

6.6 Drone list

A drone list form was implemented to display all connected drones and their status, including controls for changing system IDs and for requesting parameters. This form was made to display if any of the drones had conflicting ports or system IDs or missing parameters. Multiple drones sharing the same port was an issue because then there was a high risk of losing messages that were fragmented into multiple packets. Sharing the same system ID was also something to avoid, as then the ground station would think that all the drones sharing the ID would be one and the same. For this reason, the default system ID of 1 was considered an error. Whenever a new drone was set up and connected to the ground station, it would have the ID of 1 and the software would prompt the user to give it a new ID. If the ID 1 was considered valid and the software was already connected to a drone with that ID, then it would have trouble noticing a fresh drone that connected. This window is shown in Figure 37 below.

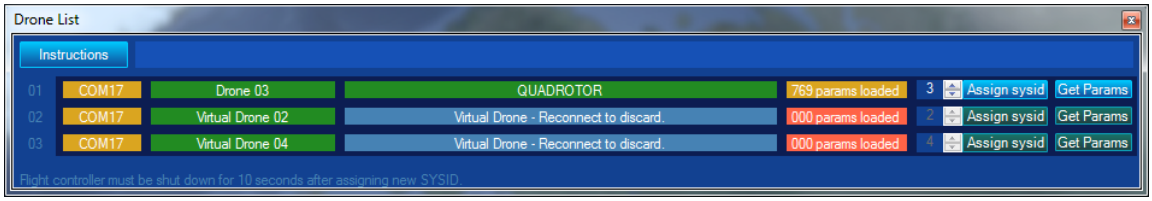

Figure 37: Drone List window

While all this information was already available scattered around the software, flying several drones often showed how important it is to be able to inspect many possible error causes from a single UI.

6.7 Usage Workflow

Implementing all the above modifications made it possible to fly routes with a formation of drones. The workflow went as follows:

1. Connect to the leader drone over UDP
2. Open 'Formation Control' form
3. Perform UDP scan to connect other drones
4. Adjust formation parameters and ordering of drones if necessary
5. Plan a mission and write it to the leader drone
6. Arm all drones and wait for all of them to be armed
7. Takeoff all drones and wait for them to get some altitude
8. Activate swarming and wait for the follower drones to find their spots in the formation
9. Start mission

6.8 Utility Tools

Early on during development, setting up connections to the drones became a repetitive and time-consuming chore. Because of this, a small tool was developed to expediate this process. All drone names, IP addresses and ports were hardcoded into the tool. When launched, it would ping all known addresses and display the connection status and latency. It could also be set to automatically refresh the connection status by sending pings periodically. This tool made it possible to monitor how long it would take for the drone to come online, to see when drones would drop from the network and the latency display even allowed to find out when drones accidentally connected to the incorrect network. Although this tool was built separately on the side, it still managed to be a significant help in setting up and debugging the drones. Although it would have been possible to develop this functionality directly into the GCS, it was thought that it is better to make it a separate application to avoid increasing the complexity of the GCS. Also, in the end the drones will be developed to a point where they can autonomously resolve most setup issues, making these features obsolete. This tool is shown in Figure 38 below.

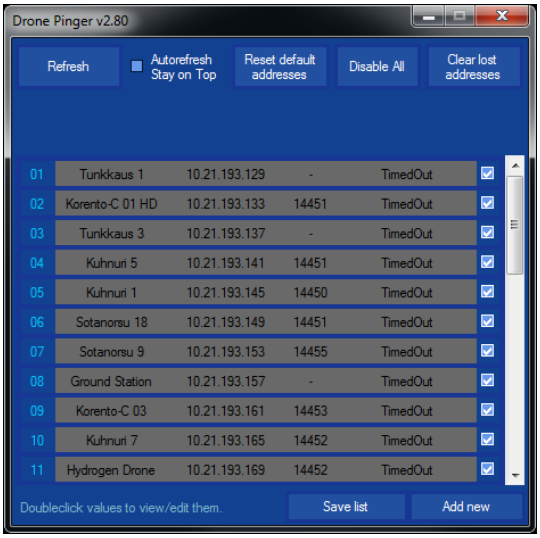

Figure 38: A tool developed for debugging connections.

Later on, during the project, it became common to have to run or restart scripts running on the drone's onboard computer to fix issues with video streams or communications. This was a task that required the user to connect to the drone over SSH, log in and type several commands. Another tool was developed to automate this process so that the setup of the drones before test flights would be much smoother. At the time, it was more convenient to build a fresh application around the SSH automation feature than to try expanding the previous tool, which is why it became its own entity. This tool proved very useful when the drones' onboard software was still under development. This application is shown in Figure 39 below.

Figure 39: Tool developed for running and restarting scripts on the onboard computer

6.9 First Tests and Learnings

The modified software was tested on multiple occasions to ensure correct behavior. The tests were carried out in good weather conditions outdoors with up to three drones. Additional pictures from the test flights can be seen in Appendix A – Mission footage.

During one of our early test flights, one of our drones disarmed on its own during landing at 14 meters' altitude and crashed to the ground. It was later discovered that this was an issue with the flight controller version at the time. The patch notes for the next autopilot firmware update included landing detector improvements that would prevent the vehicle from disarming itself if it was descending at a rate less than 1m/s (Mackay, 2016).

Swarm Flight

The practical flight tests quickly revealed that the leader drone would always fly several meters ahead of the others, turning a line formation into a V formation. This is shown in Figure 40 below. This asynchronous behavior was partially caused by latency. Since the waypoints were calculated as offsets of the last known location of the leader drone, the communications delay between the ground station and the drones would have to be considered. Additionally, the MAVLink waypoints have a default acceptance radius of 2 meters, which means that the waypoint is considered reached once the drone is within this distance of it. This same delay issue has also been noticed in other research, where the connections were done using traditional radio links instead of LTE (Sullivan, 2016).

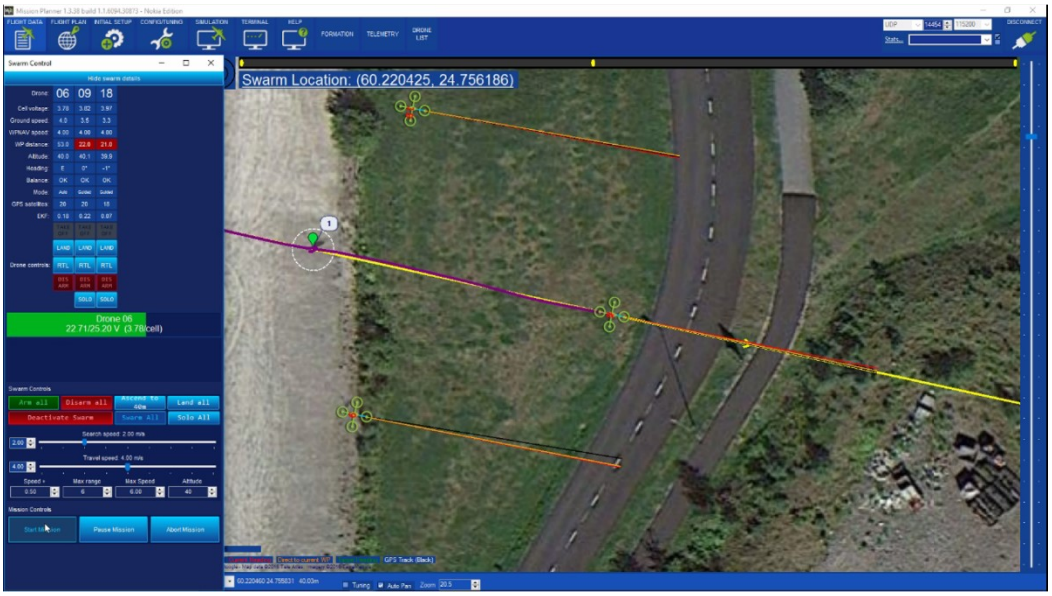

Figure 40: Screenshot of three drones being flown in formation using the prototype GCS

These delays were compensated for by offsetting the follower waypoints forwards by a few meters. Also, another feature was developed to automatically increase a follower's speed and acceleration if it fell too far behind from its waypoint. These changes enabled the drones to keep up with the formation and catch up when necessary. Options for adjusting the extra speed and the threshold for activation can be seen in the right-hand side of Figure 36 above. This was used to enable the user to fine tune the behavior of the drone formation in real time. Of course, in an ideal situation such compensation should not be required as input from the operator and it would be done automatically under the hood by the software.

However, these changes also introduced a collision risk when performing standard lawnmower pattern paths, since just before the leader drone was to reach a corner, the follower drones would still be instructed

to fly forward with given offsets. This meant that for a while the leader's and followers' paths would intersect until a new waypoint was sent. This was fixed by making sure that a follower's offset waypoint could never be set further than the next waypoint of the leader. Further, this was refined so that if the leader drone had more than one waypoint remaining, the following waypoint would also be considered to determine the maximum offset for a follower. Similar collision issues were also discovered by (Sullivan, 2016), where it was determined that a complicated algorithm would be needed to correct drone flight paths to prevent crashes. This is true for arbitrary flight paths, but in the case of a predefined lawnmower search pattern, it was sufficient to simply limit the offsets for the guided waypoints.

These fixes significantly improved the formation flight by reducing the impact of latency and by preventing the drones from crashing into each other. However, even with these fixes it was clear that sending periodic waypoint commands from a ground station would be a suboptimal and unreliable method for coordinated flights.

The swarming functionality had been implemented as a part of the window where the drone offsets are defined. This meant that if the user accidentally closed the window, the software would immediately stop giving new waypoint commands to the follower drones, leaving the leader drone to continue the mission alone. This could have been fixed by moving the functionality over to be part of the main body of the software. However, in the interest of saving time this was not implemented because it was not critical for the overall functionality of the software.

Connectivity

It was also discovered that uploading routes to multiple drones was very slow. This had two reasons: The Mavlink communication protocol required an acknowledgement for every waypoint item, which meant that any latency or packet losses would have a heavy impact on the time to complete the transaction. Also, Mission Planner was built with an architecture that supported only one ongoing transaction at a time, which meant that routes would have to be uploaded to each drone one at a time. The same issue applied to downloading drone parameters, but this was a lesser problem as it only had to be done once per drone after connecting.

6.10 Summary

Modifying Mission Planner taught a lot about what is practically needed in ground station software as well as what kind of pitfalls there are in designing the architecture. With all this experience, it was now possible to design a new control software that would be tailored for our use case.

Earlier in development there were considerations to modify the autopilot directly to perform any extra functionality that we might need on the drone side. However, it turned out to be the right choice to instead install an onboard computer for this purpose, because although it would potentially be more efficient to have all our code running directly in the autopilot, it would make it considerably more challenging to receive

updates from the open source community. Now that the autopilot was left untouched, it was possible to always update its firmware with new versions without any extra integration work.

7 Implementation of a Search and Rescue Drone System

This chapter describes the design, development and testing of a new ground control application, which was built on the learnings gained from the previous proof of concept project described in chapter 6.

7.1 Objectives and High-Level Design

The primary objectives for the new software remained the same as they were with the proof of concept project. This meant that route planning and simultaneous control for multiple drones were the main focus from the start. As a result, some features that were present in the previous stage were intentionally dropped. Most notably, the extensive drone setup, configuration and calibration features were considered low priority from the start, because those were things that could still be performed with the earlier software when needed.

Clearly, the biggest issue with the proof of concept application was the way it handled connections and interactions with drones. Because that software was apparently designed for controlling only a single drone, its original design did not account for a scenario where commands and other interactions should have been performed simultaneously to several drones. Instead, the software would handle one interaction at a time, waiting for acknowledgements from the drone and even lock up the user interface during long transactions like loading parameters and routes.

In order to prevent these issues, the new software was required to have stateless, simplified and separate communications. The separation is achieved by having each communication link run in their own dedicated thread in the background. This will ensure that even long transactions will not block communications with other drones or cause the software UI to become unresponsive. The statelessness is achieved by considering input and output as separate entities. This means that the software will never get stuck waiting for a response to a command and any acknowledgement checking functionality can later be built on top as a separate optional component.

7.2 Approach comparison

Before starting to work on the new GCS application, it was first necessary to decide which kind of application made most sense to create. Several existing ground stations were researched in chapter 4 and the following types of applications seemed like feasible starting points:

- Windows Forms application (such as Mission Planner)
- Web application (such as Litchi)
- Unity application (such as UgCS)
- Qt application (such as QGroundControl)
- Mobile application (such as Tower)

These approaches are compared in Table 4 below. Being built with C# is considered beneficial because it means that experience gained from the proof of concept project will help speed up the development.

Table 4: Comparison of application types for GCS development in the context of this thesis

Application type	**Pros**	**Cons**
Windows Forms	• Built with C#	• Windows only
Web	• Accessible with any web browser • Application doesn't necessarily have to be installed on the GCS device	• Different development environment • Would have to be built from scratch • Not as fast or responsive as a desktop application • Might run into browser incompatibilities
Unity	• Built with C# • Multiplatform • Convenient for 3D development	• Different development environment
Qt	• Multiplatform • Convenient for GUI development	• Different development environment • Would have to be built from scratch
Mobile	• GCS would be light to carry and easy to transport	• Different development environment • Would have to be built from scratch • Limited screen space and control accuracy of mobile devices

A web application was considered unfeasible, because from earlier experience, such applications tend to be considerably less responsive compared to desktop applications, and they may run into issues when the wrong browser type or version is used. Also, the development of a web application would be very different from the type of development done on the proof of concept project, which means that prior experience could not be utilized to its fullest. Mobile applications were also considered unfeasible mostly because mobile devices, while being light and easy to transport, do not have the screen space and the kind of accurate and fast controls that a GCS for a drone fleet would require.

Qt framework and the Unity engine were interesting options. Unity would make it convenient to integrate 3D graphics to the application, whereas Qt would make Graphical User Interface (GUI) development more straightforward. Both applications would also enable deployment to multiple platforms. However, working with Qt would require learning the framework, which would make it slow to start the project. Unity, although also supporting C#, would still require getting used to the editor, and thus benefit less from the learnings gained from the proof of concept project.

In the interest of fast development, the new GCS was decided to be created as a windows forms application. This meant that the knowledge and familiarity gained from working on the proof of concept project could be utilized to its fullest to avoid pitfalls and to ensure a fast and smooth start.

7.3 Methodology

The work started by putting together GMap.NET (Radioman, 2008) to provide the online maps and Mavlink generator output (Meier, 2009) to provide the message definitions necessary for communicating with the autopilot. The actual workflow during this project was similar to the earlier prototype project, because the same programming language and development environment were used in both. This meant that the development, testing, debugging and work on roadmaps was done as previously. Also, the utility tools developed during the prototype phase continued to prove useful throughout the rest of the project.

Because developing a GCS application is bound to become a large project, simplicity and modularity were high priorities from the start to ensure fast development and minimal time spent dealing with bugs. This approach had a major impact on design decisions and feature priorization.

7.4 User Interface Development

The User Interface (UI) creation was started by first filling the main application form with a map and by creating a separate window for controls. The software was planned to have different states dedicated to different types of functionality. There would be route planning mode, where the user could create waypoints on the map to create routes for the selected drones. Area planning mode was supposed to enable the user to specify a region on the map, which could then be used to automatically generate routes for selected drones or to specify the area as a no-flight zone. Mission control mode was meant for controlling drones on preplanned missions and guided control mode was meant for allowing the user to control the drones in real time by assigning waypoints to selected drones directly by clicking on the map. Settings mode was meant to

be a separate screen for various configuration options for both the software itself and the connected drones. A screenshot of this early design is shown in Figure 41 below.

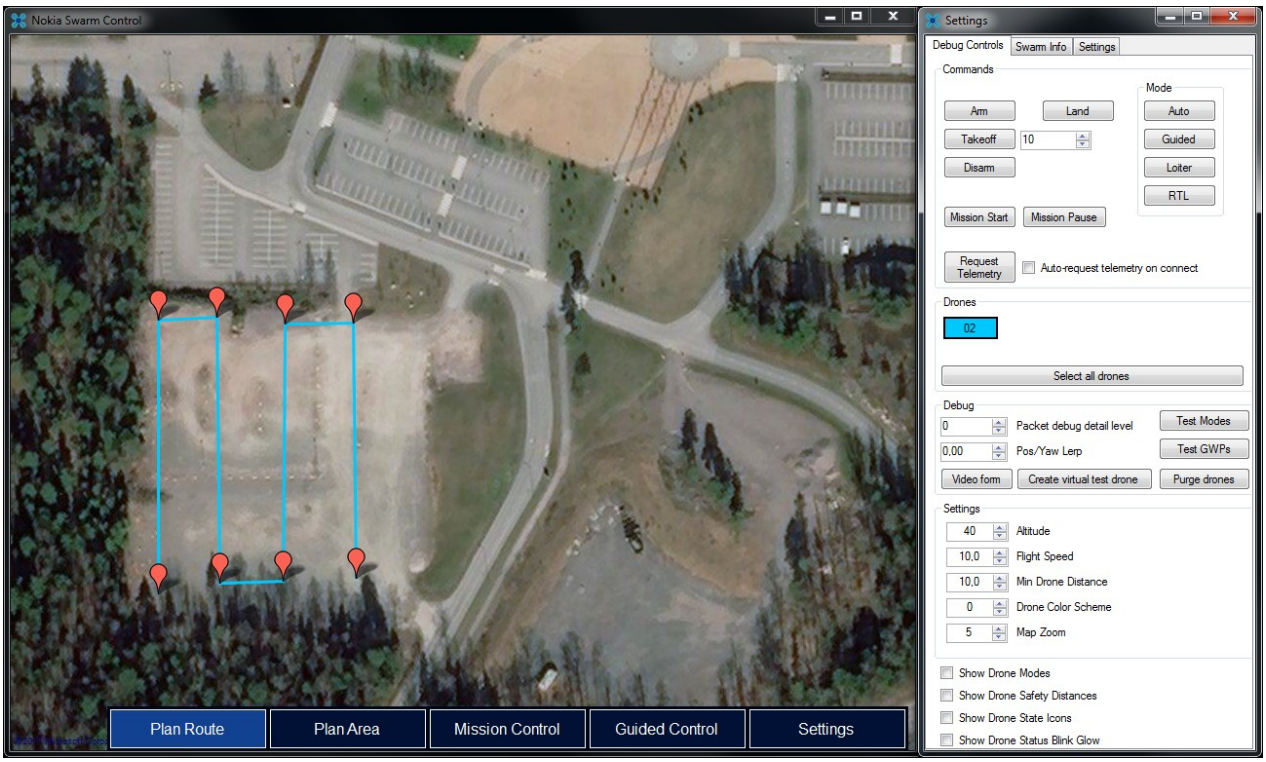

Figure 41: A very early development screenshot of the new ground station software

The separate control window (labeled as 'Settings' on the right side of the figure above) was created as a fast tool to speed up development. Any new functionality could be added as a button on this form, leaving the main window untouched. This meant that new functionality could be tested with minimal time investment and a proper user interface could later be built on a clean slate.

One of the first improvements to the main window was the introduction of a top and bottom bar elements above and below the map. The bottom bar would contain the mode buttons that were previously overlaid directly on top of the map. The top bar featured a fleet list on the left with icons for all connected drones. The icon contained a system identifier (ID) number for identifying the drone, a body color for status (green for ok, yellow for warning, red for error, gray for disconnected) and a background color for showing the selection status. This simple UI component enabled the user to immediately see the overall status of all connected drones and to change their drone selection regardless of whether the drone was visible on the current map view. Clicking the 'Fleet'-label would instantly select or deselect all drones. The middle portion of this top bar was reserved for notification messages, which were meant to display generic feedback to the user about what is happening, such as drones connecting, disconnecting, having errors and so on. These additions are shown in the screenshot below.

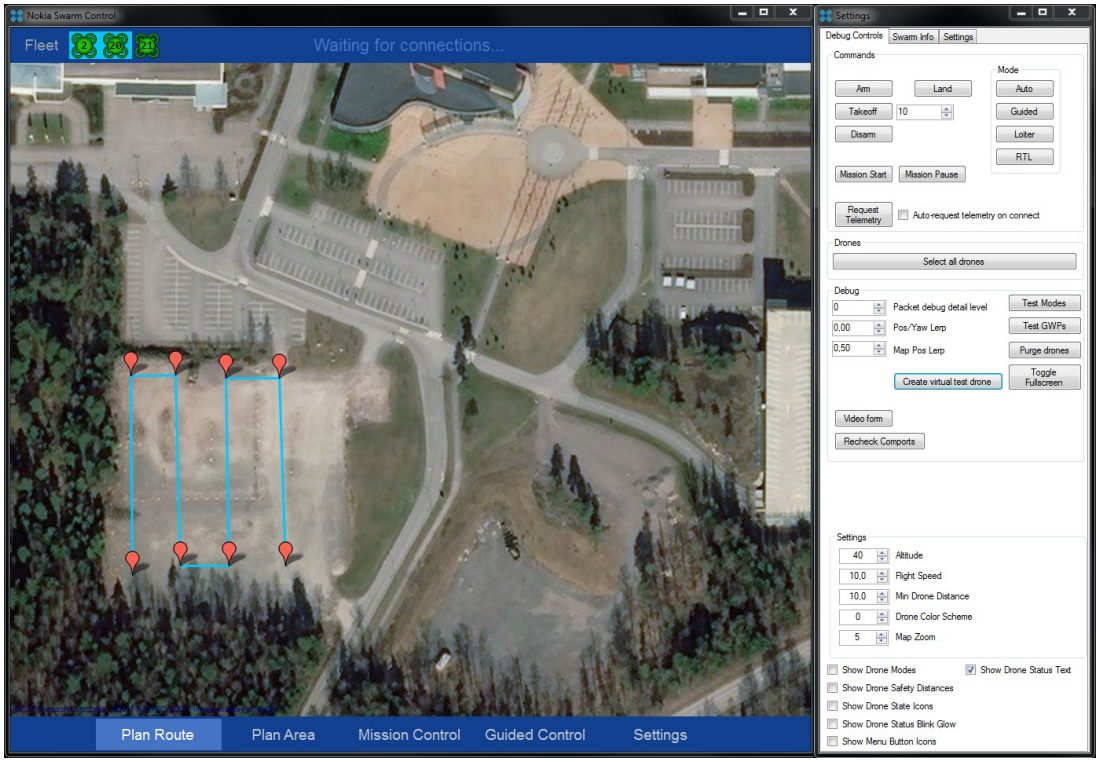

Figure 42: Early version of the interface with the main window divided into three rows

Next, the drone map markers were introduced to the software. The markers were designed to have a quadcopter shape with distinct borders to make them very clear to distinguish from the satellite maps. The borders would be rendered dark for unselected drones and bright for selected drones. As with the fleet icons on the top bar, the map markers also had the drone ID number displayed in the middle for identification. Also, at this point the route display was adjusted. A dark outline was added to ensure the lines were visible even in bright backgrounds, the waypoint markers were labeled with index numbers and the first and last waypoint were distinctly colored with green and red respectively. These changes can be seen in Figure 43 below.

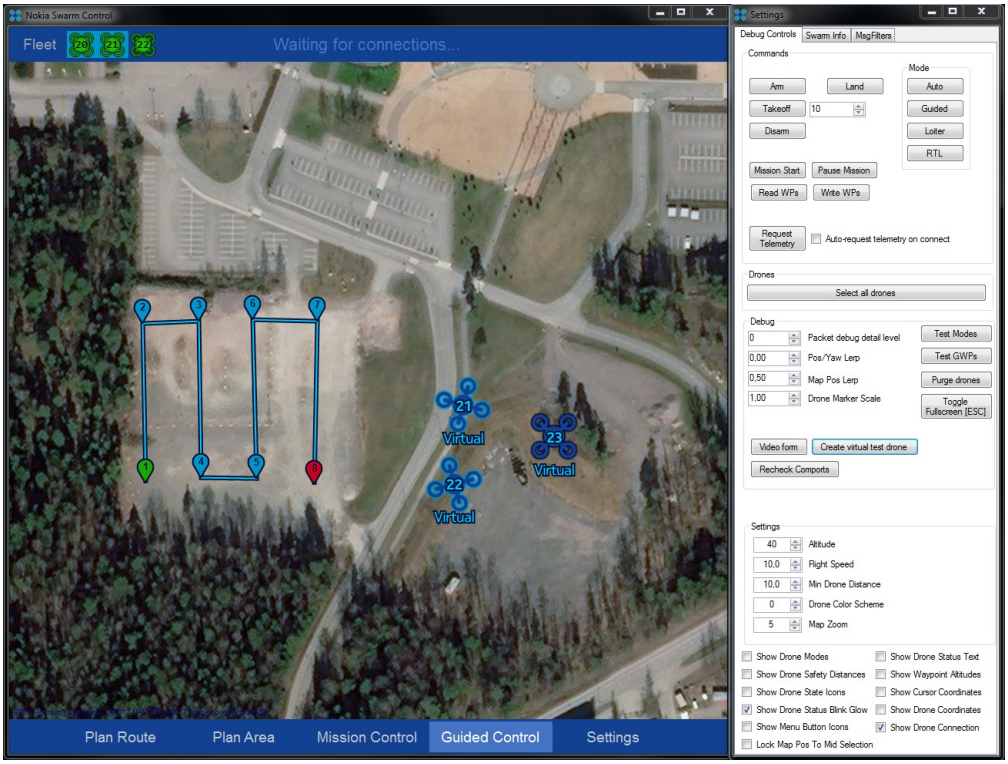

Figure 43: Introduction of drone map markers and updated route visuals

Furthermore, it was quickly noticed that simply displaying the drone markers at the last location reported by the drone would cause the marker to jump, which did not give an impression of the drone moving at a smooth speed. This was fixed by having the drone marker move towards the last reported position using linear interpolation. The same approach was used to smooth the rotation of the marker.

Having the drone markers animated soon inspired to also animate the routes. The direction and flow of the mission could be intuitively visualized by having an animated dashed line run through the route. This meant that the user could look at any portion of a route and immediately see where the drone would be approaching and where it would be heading thanks to the animation. The initial idea was to also synchronize the animation speed with drone flight speed for a more accurate preview, but this was dropped for not having enough practical benefit to justify the time required to synchronize the animation to map zoom levels and to split up the route into segments to account for waypoints followed by speed change commands. A faint line was also drawn from the drone marker to the first waypoint of the route to make it clear which route belonged to which drone. These changes are shown in Figure 44 below.

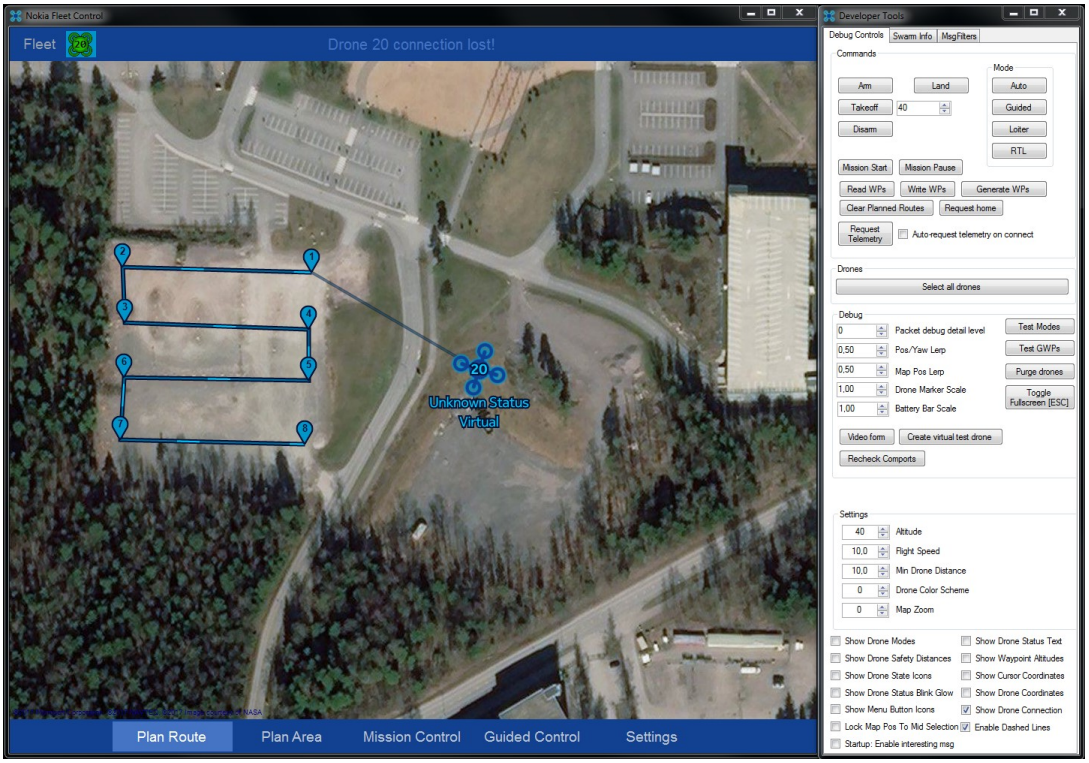

Figure 44: Routes with animated dashes

The next improvement regarding routes was to separately display routes that were already written to a drones' onboard autopilot. This meant that the written route would be displayed as a transparent shadow in the route planning mode to make it distinct from planned routes that the user could edit. In control mode, planned routes would be hidden and the written routes would be displayed with solid colors to make it clear for the user that the drone would follow this route if given the 'start mission'-command.

Another addition was the battery indicator bar above the drone marker, which would display the battery percentage reported by the autopilot. The shape of the indicator was made to resemble a 'health bar', which is a common concept in video games to make it intuitive for the user.

The bottom bar was also worked further, the buttons for unimplemented modes like area planning and settings were disabled and grayed out and icons were created next to the buttons. These changes are shown in Figure 45 below.

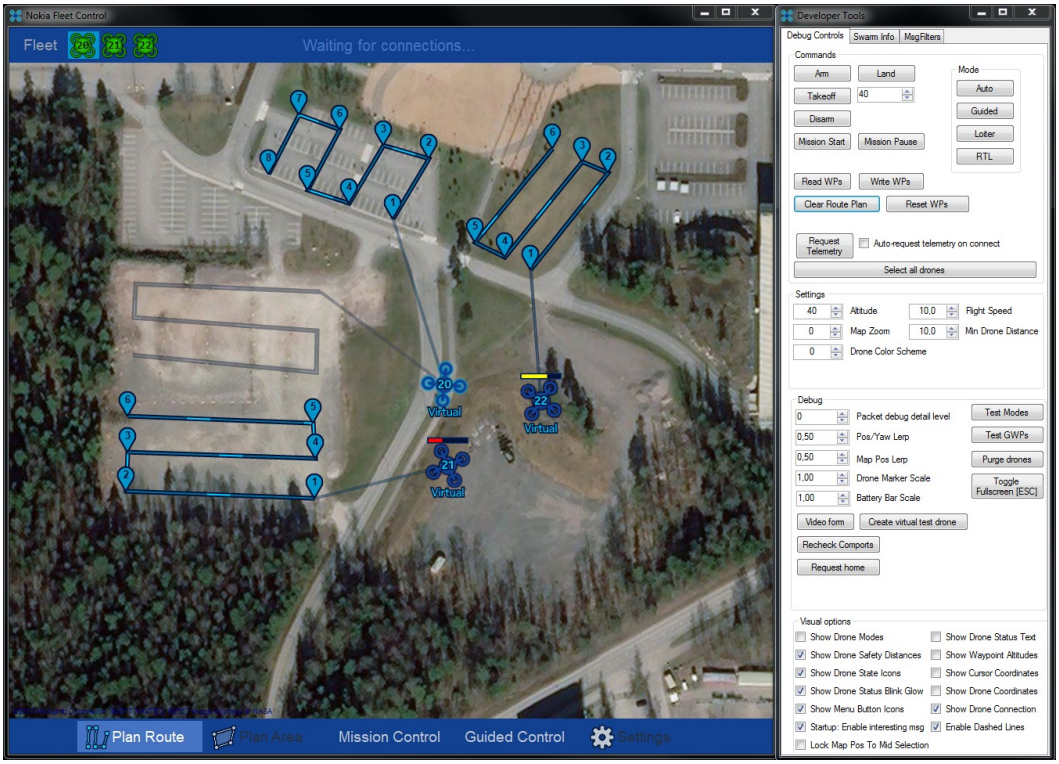

Figure 45: Battery indicators and written routes

Next, trails were implemented for the drones. Each drone marker was now followed by a yellow line that denoted the path it had flown. Also, the battery indicator was worked further by segmenting it with lines to 10 percent portions that would make it clearer for the user to estimate the value. The bar was also set to blink when it went red to capture the user's attention better. The drone markers were now also given transparent field of view cone visuals to display the facing direction of the drone. These cones were meant to be later expanded to accurately display the camera boundaries similar to how it was done earlier in Figure 34. These changes are shown in Figure 46 below.

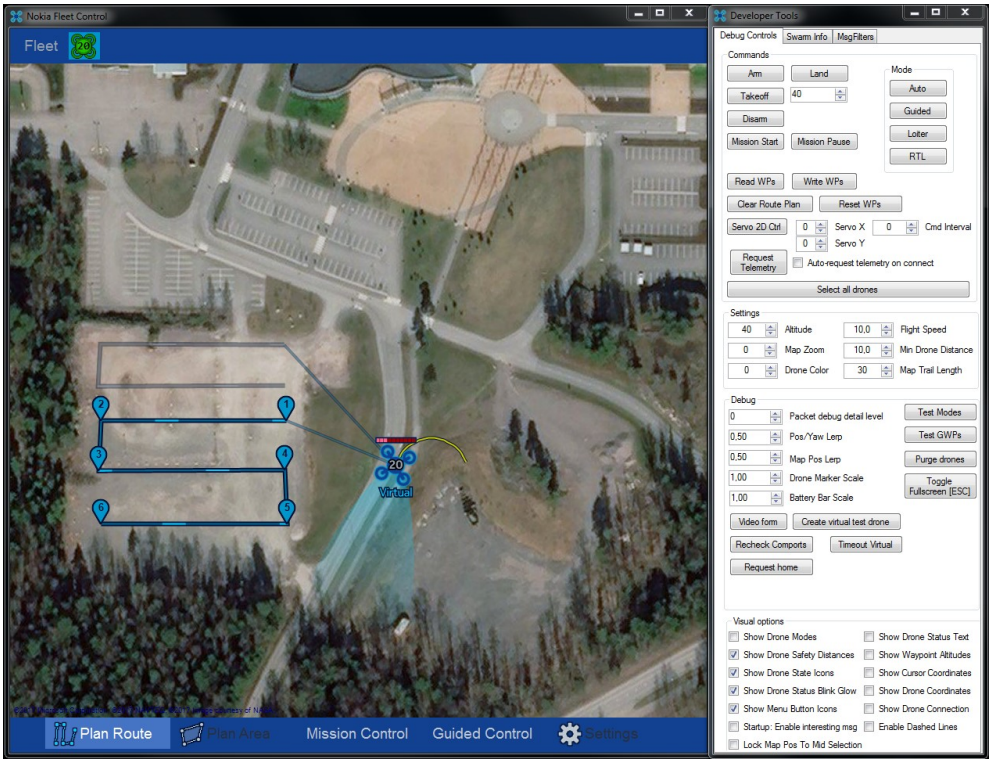

Figure 46: Drone trails and heading field of view cones

Up to this point, routes had been simply lists of waypoints, even though the Mavlink protocol supported several commands and parameters that could be included in missions. To enable the user to have access to these commands, a route configuration form was created. This form listed all waypoints of the route and enabled the user to adjust the commands and parameters to customize the route further. Additionally, the top portion of the configuration form was dedicated to displaying an overview of the route, including information about the assigned drone, flight speed, total distance, estimated flight time and battery usage.

The route visualization on the map was also improved by displaying the distances between waypoints.

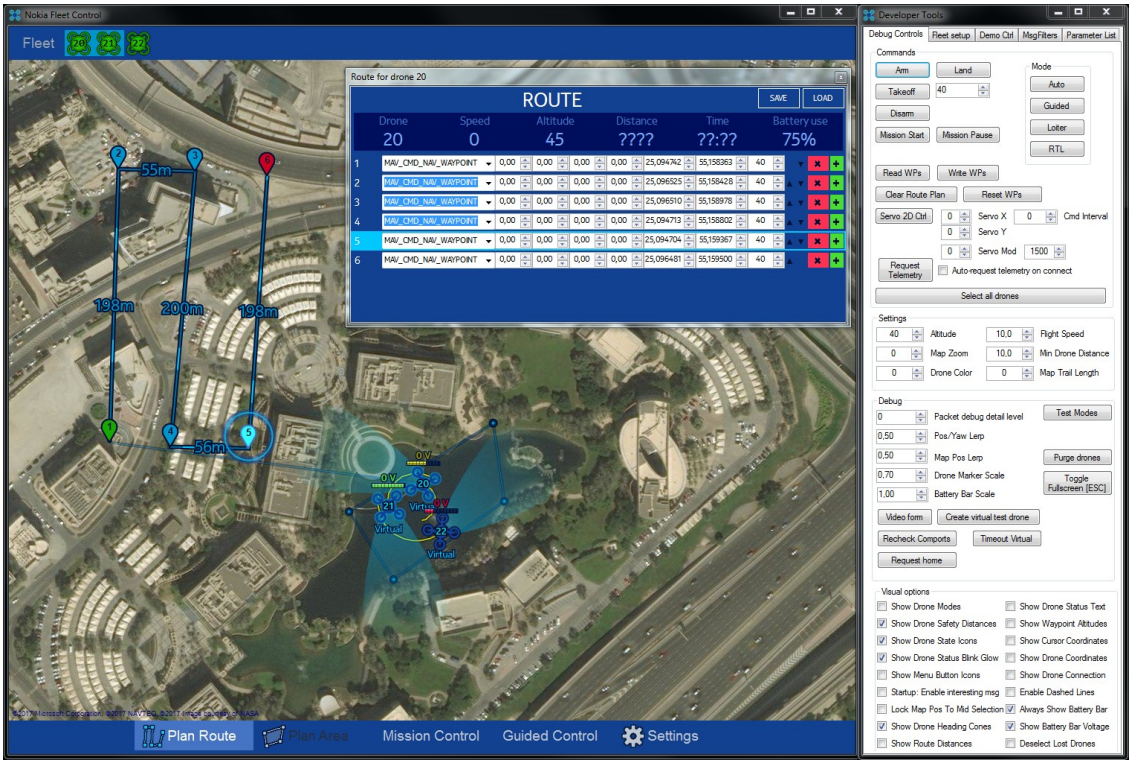

Figure 47: Route configuration and route segment distances

Virtual drones were implemented to enable testing the software rapidly without having to connect a real drone. This meant that new drone instances could be created in the software instantly with a single button press, which made it very fast to test basic functionality. Later, Software In The Loop (SITL) simulators were used to ensure correct behavior. SITL produces an accurate simulation of an ArduPilot multicopter, which makes it possible to test how the controls and missions created by the software would work with a real drone. However, the downside with SITL is that it takes a long time to start up and there didn't seem to be a way to have multiple SITL simulated drones running simultaneously. For these reasons, the original virtual drones remained a valuable asset throughout development despite their simplicity.

At one point in development, the bottom bar was consolidated to display only three buttons: setup, plan and control. The icons were also discarded for the sake of simplicity and to make the interface seem less cluttered. The fleet drone markers were moved from the top bar to be on top of the map, so that it could be scaled up for a large number of drones. Also, a separate form was made for mission controls, which meant that the user could now operate the software without using the developer tools window. These changes are shown in Figure 48 below.

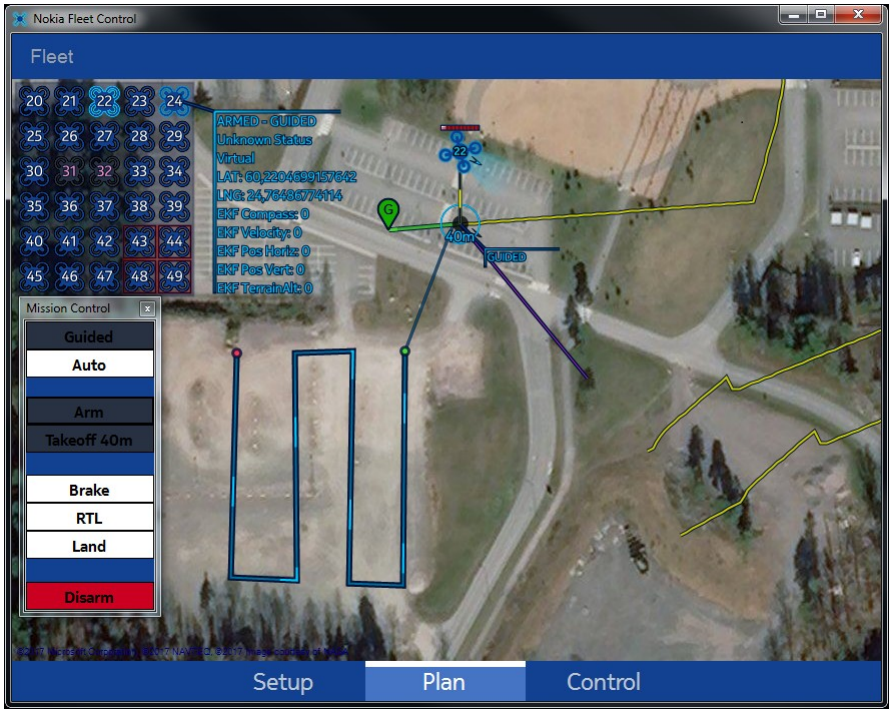

Figure 48: Updated drone fleet list, mission controls and mouseover info

Routes were set to only display their details on mouseover to prevent unnecessary clutter on the screen. Also, a feature was implemented to allow the user to insert a new waypoint in the middle of a route segment by clicking and dragging. This is similar to how some online mapping services work with their route planning or measurement tools, so the usage should be intuitive to the user. The user could also right-click any waypoint to delete it, or left click on the map to append a new waypoint to the end of the route. This is illustrated in Figure 49 below. This made working with routes much faster than before.

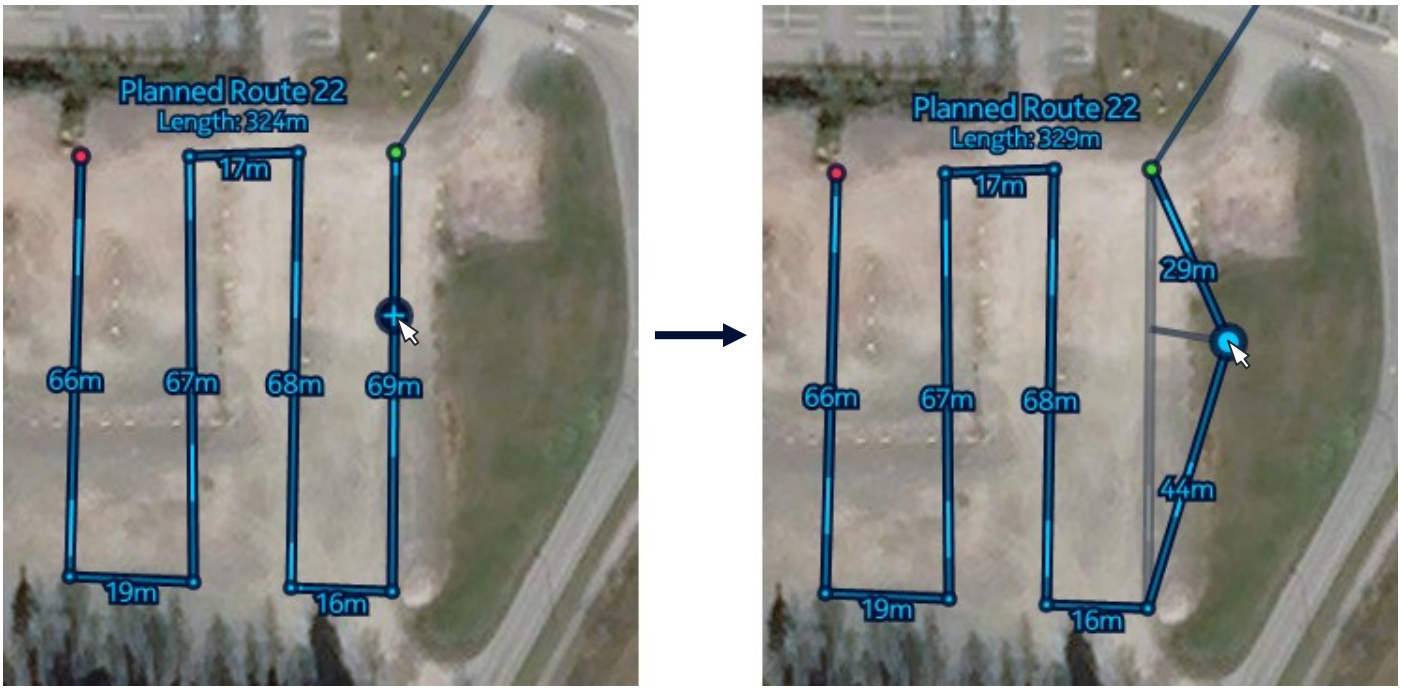

Figure 49: Route mouseover details and drag action to insert a waypoint

An experiment was made to split the interface into two components. The top and bottom bars were removed and instead, the controls were put to the left side of the screen, while the map was set to fill the right side of the screen. A message box element was implemented to display any messages coming from the drones or notifications that the system should give to the user. This was a big improvement over the earlier one-line notification that was displayed in the top bar. The previous system required messages to be shown one at a time, which was very inconvenient when several important notifications had to be sent to the user. The new message box could also be scrolled to see full history and it could be filtered to only show the most recent of any duplicate messages. The duplicate filtering was useful to prevent spam from repeated error messages, which the autopilot would send periodically as long as the error remained. These changes can be seen in Figure 50 below.

Figure 50: UI overhaul and the introduction of the message box component

7.5 Controls

Since the software is meant to be used to control multiple UAVs simultaneously, it needs to support selecting and deselecting UAVs. All of the control buttons in the software will send commands to all currently selected drones, which will make it possible to run synchronized flight missions. This also means that it is very important for the user to clearly distinguish, which drones are selected at a given time. Because of this, the always visible fleet list was one of the very early elements implemented in the user interface, as was mentioned in the previous chapter.

The basic commands the user needs to run a mission safely are the following:

- Arming the drone to make it capable of carrying out commands
- Disarming the drone to make it safe to approach and handle before it is physically powered off
- Giving a takeoff command to desired altitude to start the flight

- Giving a land or return to launch command to end the flight
- Pausing the drone to make it hover in place until further orders are given
- Starting, stopping, continuing and aborting a preset flight mission

UI buttons for these commands were shown in Figure 48 above.

There are also commands that require the user to specify a location on the map by clicking. Guided waypoints are commands that are used to send the drone to a target location immediately. They can be used to perform ad-hoc missions without having to plan and upload a route beforehand. The drones can also be given a Region of Interest (ROI) point, which the drone will face during flight. This is useful if we want to ensure that the camera of the drone is pointing at a specific location even when it is not flying directly towards it.

The ROI functionality was later expanded so that the user could set a drone as another drone's point of interest, which would mean that it would effectively always look at the other drone. This was implemented by having the GCS send ROI commands periodically to the chosen drone with the target drone's coordinates as the parameters. This feature was useful in scenarios where one drone was used to supervise the progress of other drones in order to get an overview of the situation.

The guided waypoint command was extended to work with multiple drones simultaneously, which meant that the current formation of the selected drones was ordered to move to the target location. With this approach, the formation was never rotated so that collisions resulting from crossing waypoint lines could be prevented.

7.6 Communications

The wireless communications between GCS and the drones were implemented over UDP, because the vast majority of the traffic is telemetry information sent by the drone. The GCS is interested in the most recent status of the drone, which means that if a telemetry packet is lost, it is preferable to receive the next packet on time, instead of resending outdated information. For this reason, TCP connections were not implemented, because it might have caused reduced performance in poor network conditions.

Ideally, TCP would be used to reliably send any commands to the drone, while telemetry would go over UDP. However, for the sake of simplicity, all traffic was built on UDP, and in the rare case that the drone was to fail in receiving a command, the user would simply have to resend it.

In order to reduce the time it takes to upload missions to individual drones, I experimented with sending all waypoint messages in a row without waiting for acknowledgements. This seemed to work fine, so to further optimize the communications, I tried to concatenate all waypoint messages into a single UDP packet. This approach reduces the load on the network and removes the chance of waypoint packets arriving in the wrong order. Since the autopilot reads incoming Mavlink data one byte at a time, this method worked fine until the messages became too long. Testing revealed that routes that were more than roughly 30 waypoints in length would not make it to the autopilot intact, which meant that long routes would have to be

transmitted the original way with the latency issues. This research lead into a patent application filed by Nokia.

To overcome the route size limit of the previous solution, a communications link using Google Protocol Buffers (gRPC) was developed between the ground station and the onboard computer. With this approach, the route could be sent to the drone as a single gRPC message, which the onboard computer would then convert into MAVLink format and feed to the autopilot with the original method. Because the onboard computer is directly connected to the autopilot, the transaction goes smoothly as latency cannot accumulate in the same way as it did previously.

7.7 Field Tests and Final Release

To test the GCS application, a water rescue operation was performed with three drones. The first drone was equipped with HD and thermal cameras and it was tasked to first find the person in the water and then supervise the operation. The second drone was carrying a delivery system that would be used to drop a floating payload for the person in the water, while the third drone would carry a loudspeaker that would be used to communicate to the person.

The operation was carried out by sending the camera-equipped drone to perform a low altitude lawnmower pattern search of the water area and having the two other drones stand by near the shore. This scenario can be seen in Figure 51 below.

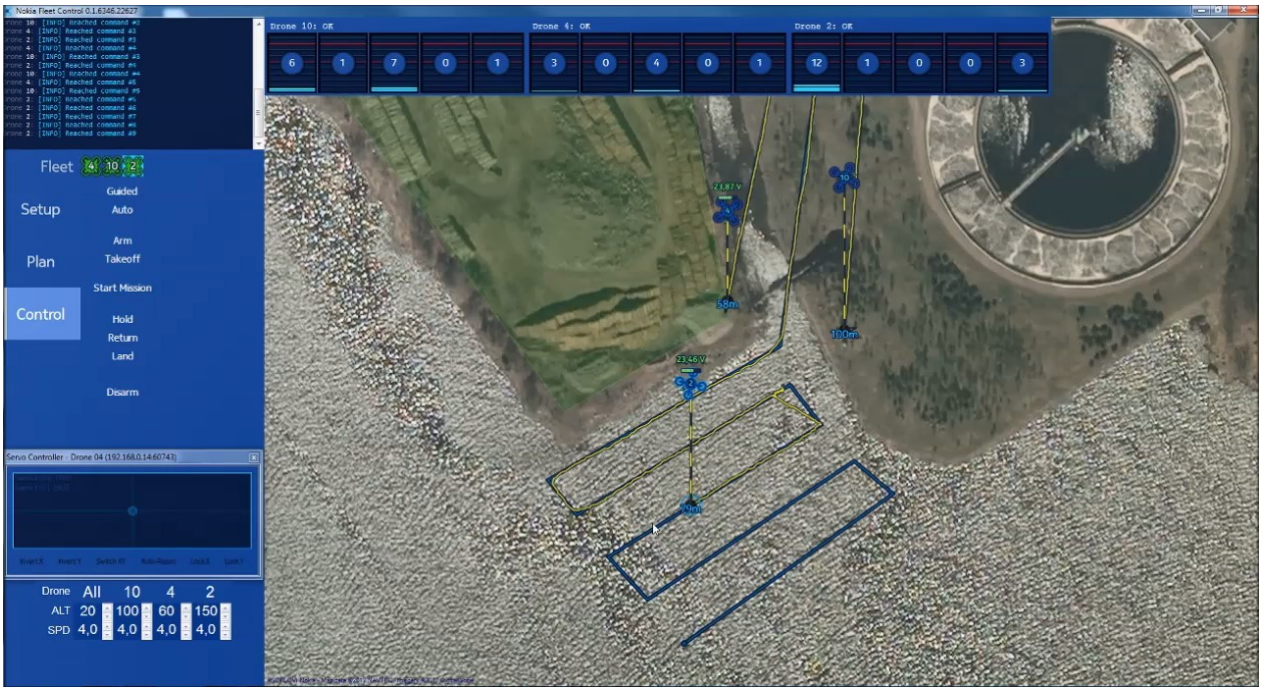

Figure 51: The GCS used to perform a water search for a missing person

It turned out that performing a low altitude scan of the area straight away is not necessarily the best approach. It is instead better to first bring a drone to high altitude and get a general overview of the area, because doing so might reveal points of interest that should be investigated immediately. A screenshot of the thermal and HD video streams is shown in Figure 52 below. It was found that a person in cold water is

practically invisible to the thermal camera that was used. In the HD camera view, it is possible to see the person in a red drysuit, the white floating payload that was dropped near him, and the third drone hovering nearby. However, in the thermal view, only the drone is visible.

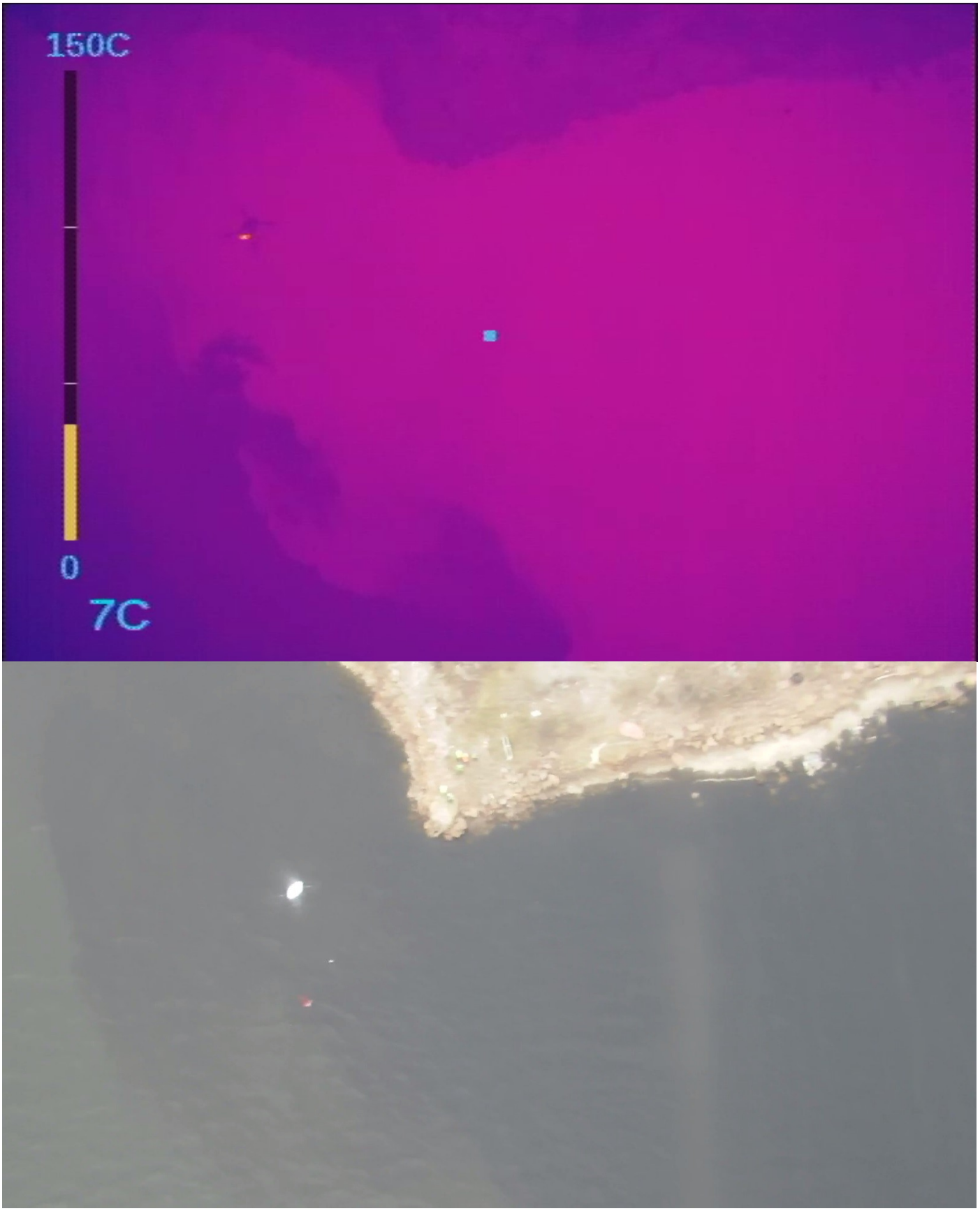

Figure 52: Screenshots of Thermal and HD video streams from the water rescue scenario.

During this trial, we also learned that in practice a search mission will have to be flexible and modifiable, as professionals on site prefer having direct control on the progression of the operation. It was also verified that having a clear user interface and a smooth workflow is critical in being able to modify missions in real-time fast and reliably. Similar observations have been made in field tests in earlier research (Goodrich, et al., 2008). Also, it has been found out that actual search operations are so varied and irregular that they cannot be precisely formulated mathematically, and that only crude approximations can be made of the optimal allocation of efforts (Frost & Stone, 2001).

Two aspects of the UI were found to be detrimental to the usage of the software. First, the flight trails of drones made a lot of clutter on the screen. To fix this, a toggle button was implemented to show or hide the trails, and another button was added that enabled the user to clean up the trail of a drone, which was a useful function to use between different flight missions. The other issue was with the general layout, where there wasn't enough vertical space to display long messages in the message box and the buttons for setup, plan and control modes were positioned in an inconvenient place. This was remedied by re-introducing the top bar to the UI. The top bar now had the buttons for application mode, the fleet list, and the message box. The left-hand side column housed all of the mode-specific buttons of the currently selected mode. These changes are shown in Figure 53 below.

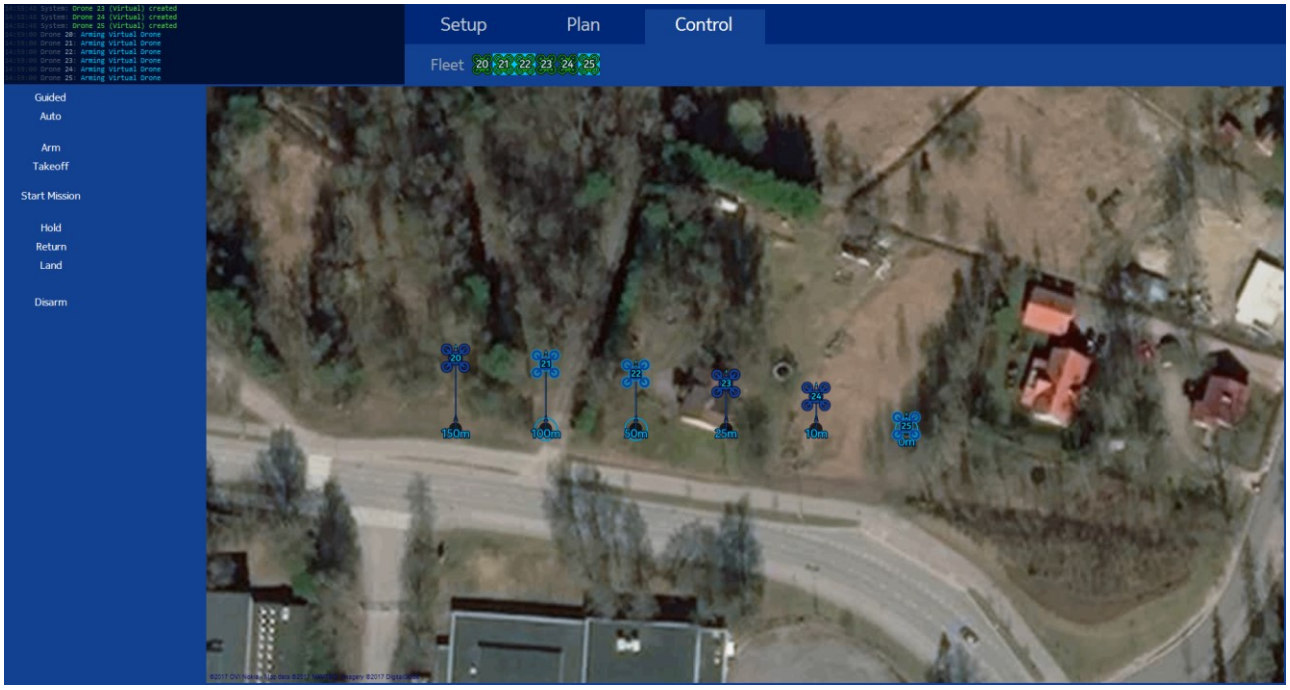

Figure 53: Re-introduction of the top bar UI element

After some additional testing, more adjustments were made to the UI. Most importantly, an emergency stop button was added near the top right corner of the software. Pressing this button would immediately pause all connected drones in their current location, giving the user a one-click solution to resolve any dangerous looking situation. With previous UI, the user would have first had to select the drones, then made sure he was in control mode, and finally given the stop command. Also, a column was added to the right-hand side for map controls. Although the map could be scrolled with mouse scrollwheel, in field conditions it was

sometimes necessary to use the software with a trackpad, so a UI zoom slider was necessary. Also, buttons were added for instantly moving the map view to the currently selected drones, and filtering buttons to show, hide and clear the drone trails. Also, the trails and route plans were now displayed with a different color for each drone to make it easy to distinguish them from each other. Also, a GCS-based collision prevention system was made. This system simply evaluated the distances and headings of all connected drones, and showed warnings if drones got too close to each other and automatically sent a pause command to any drone that was about to collide with another. Three toggle buttons were added to the UI. One to toggle the visibility of planned missions and mission controls, one to toggle the ability to assign manual waypoints by clicking the map, and one to toggle the automatic collision prevention on and off. These changes can be seen in Figure 54 below.

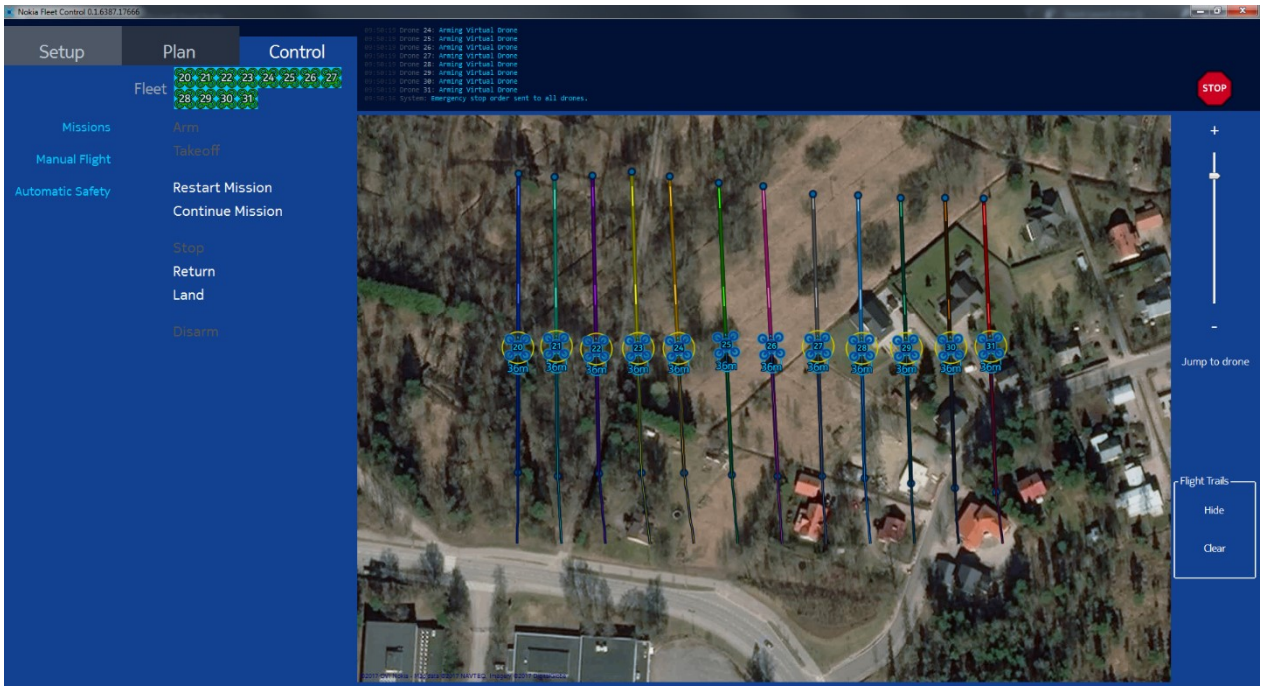

Figure 54: Emergency stop button, map controls and other UI adjustments

8 Conclusion and Future Work

The purpose of this thesis was to first find out what requirements there are for a GCS application suitable for controlling a fleet of drones for a search and rescue use case, and then to discover how to design and develop such an application. To find the requirements, a literary review was conducted on UAVs, SAR practices and existing GCS applications. Then, a proof of concept GCS was built on top of an existing open source application. The design and development of a new GCS was then done based on all the previous learnings and the application was tested in field tests to discover how well it actually met the requirements.

It was found that LiPo-powered multicopters are maneuverable and reliable enough to make them the preferred choice of UAV for SAR scenarios, which can be very dynamic in nature as information is gained over time. It was also found that there are many search and rescue patterns and practices, and that the

missions should be planned based on all available information for the given situation, including knowledge of the environment, weather, any past operations in the area and the profile of the missing person. It was also noted that SAR professionals prefer to be in direct control of the mission progression, which means that the GCS must be capable of adjusting plans dynamically on the fly. Many existing GCS software were investigated, and it was found that a GCS can be built in multiple ways and on multiple platforms. It was also noted that most existing software is primarily intended for using a single drone and many of the features provided by these applications are primarily geared towards industries or hobbyists. A proof of concept project was built on top of an existing open source GCS application and this yielded much practical knowledge that proved valuable when designing the architecture and setting the priorities for a new GCS application. The new GCS was developed and the work on optimizing communications lead into a filed patent application. Field tests with the new GCS revealed that the software could be used to control multiple drones simultaneously in a search and rescue operation. Several interface improvements were discovered during the field test and then implemented into the software.

The main learnings were that having a clear UI is critical for controlling multiple drones, and that for SAR scenarios, creating or generating a single flight mission is not enough. The GCS must have the flexibility to adjust plans ad-hoc according to the situation.

8.1 Future Work

Next steps would be to perform more scalability related testing and development. As the number of connected drones increases, the wireless communications should be dynamically optimized by adjusting telemetry rates depending on whether drones are visible on the current map screen or selected by the user. For large fleets in a big operation it would also be beneficial to split the fleet between multiple ground stations. Then the fleet monitoring and control tasks would be shared by multiple UAV operators. However, it would also require communications between ground stations to display other stations' drones and route plans for collision avoidance and coordination.

Further on, features should be incorporated to move the user interaction away from precise route planning and towards describing the scenario and target area on a higher level. For example, area details such as elevation, vegetation, buildings and weather could be queried from online services. These details could first be used to assist the user in mission planning and later algorithms could be developed to enable the software to generate optimal flight plans automatically.

Ideally, the system would be advanced enough to perform the entire SAR operation autonomously with minimal human input. However, developing a fully autonomous system is a very time-consuming and complex task and it would take a long time for automatically generated and monitored flight plans to match the performance of operations directed by professionals in real time. For this reason, the best approach for future development seems to be focusing on improving the basic functionality of the GCS and making improvements that are immediately useful for the professionals. This way the software will always be useful in practical scenarios, whereas if all development is focused directly on the end goal of creating a fully autonomous system from the start, then the system will be of no practical value until it manages to surpass

the efficiency of traditional human-operated missions. In other words, it is better to use the case of professional-driven search as a starting point, and seek to improve it step by step towards an autonomous solution, so that the developed methods are of assistance to the efforts instead of a substitute.

9 References

3D Robotics, Inc, 2017. *Site Scan Field App.* [Online]
Available at: https://3dr.com/enterprise/features/
[Accessed 20 October 2017].

Adams, A. L. et al., 2007. Search Is a Time-Critical Event: When Search and Rescue Missions May Become Futile. *Wilderness & Environmental Medicine,* 18(2), pp. 95-101.

Adams, J. A. et al., 2007. *Camera-Equipped Mini UAVs for Wilderness Search Support: Task Analysis and Lessons from Field Trials,* s.l.: Byuchmi Technical Report.

Aguilar, A. & Benítez, A., 2004. *Nuclear, Biological, Chemical (NBC) Reconnaissance Vehicles: Current Achievements and Technology Trends at SPA.* Madrid, Servicios Proyectos Avanzados Sa Madrid (Spain).

Ahmad, W., 2016. *Distributed Navigation of Multi-Robot Systems for Sensing Coverage,* New South Wales: The University of New South Wales.

Anderson, C., 2008. *BlimpDuino home page.* [Online]
Available at: http://diydrones.com/profiles/blog/show?id=705844%3ABlogPost%3A44817
[Accessed 24 October 2017].

ArduPilot Dev Team, 2016. *APM Planner 2.* [Online]
Available at: http://ardupilot.org/planner2/
[Accessed 19 October 2017].

ArduPilot Dev Team, 2016. *Choosing a Ground Station.* [Online]
Available at: http://ardupilot.org/copter/docs/common-choosing-a-ground-station.html
[Accessed 27 October 2017].

ArduPilot Dev Team, 2016. *Swarming.* [Online]
Available at: http://ardupilot.org/planner/docs/swarming.html
[Accessed 9 October 2017].

ArduPilot Dev Team, 2017. *Rotor Craft types.* [Online]
Available at: http://ardupilot.org/copter/index.html#rotor-craft-types
[Accessed 23 October 2017].

Arnold, K. P., 2016. The UAV Ground Control Station: Types, Components, Safety, Redundancy, and Future Applications.. *International Journal of Unmanned Systems Engineering,* 4(1), pp. 37-50.

Astély, D. et al., 2009. LTE: The Evolution of Mobile Broadband. *IEEE Communications Magazine,* 47(4).

Bayat, B. et al., 2017. *Environmental Monitoring using Autonomous Vehicles: A Survey of Recent Searching Techniques.* Lausanne, Elsevier.

BBC, 2016. *Hydrogen-powered drone takes flight.* [Online]
Available at: http://www.bbc.com/news/av/technology-35890486/hydrogen-powered-drone-takes-flight
[Accessed 19 October 2017].

Bento, M. D. F., 2008. *Unmanned Aerial Vehicles: An Overview.* [Online]
Available at: http://www.insidegnss.com/auto/janfeb08-wp.pdf
[Accessed 1 August 2017].

Bertuccelli, L., Choi, H.-L., Cho, P. & How, J., 2009. *Real-time multi-UAV task assignment in dynamic and uncertain environments.* Chicago, AIAA.

Birnbaum, Z. et al., 2015. Unmanned Aerial Vehicle Security Using Recursive Parameter Estimation. *Journal of Intelligent & Robotic Systems,* 84(1-4), p. 107–120.

Bor-Yaliniz, I. R., El-Keyi, A. & Yanikomeroglu, H., 2016. *Efficient 3-D Placement of an Aerial Base Station in Next Generation Cellular Networks.* Kuala Lumpur, Malaysia, IEEE.

Boutilier, J. J. et al., 2016. Quantifying the Value of Drone-Delivered Automated External Defibrillators in Cardiac Arrest Response. *Circulation,* 134(Suppl 1).

Brass, K., 2015. *The Architecture of UgCS.* [Online]
Available at: http://diydrones.com/profiles/blogs/the-architecture-of-ugcs
[Accessed 25 October 2017].

Bürkle, A., Segor, F., Kollmann, M. & Schönbein, R., 2010. Universal Ground Control Station for Heterogeneous Sensors. *International Journal on Advances in Telecommunications,* 3(3), pp. 152-161.

Chapman, A., 2016. *Types of Drones: Multi-Rotor vs Fixed-Wing vs Single Rotor vs Hybrid VTOL.* [Online]
Available at: https://www.auav.com.au/articles/drone-types/
[Accessed 28 July 2017].

Choi, H., Geeves, M., Alsalam, B. & Gonzalez, F., 2016. *Open source computer-vision based guidance system for UAVs on-board decision making.* Big Sky, MT, USA, IEEE.

Choset, H., 2000. Coverage of Known Spaces: The Boustrophedon Cellular Decomposition. *Autonomous Robots,* 9(3), pp. 247-253.

Choset, H., 2001. Coverage for robotics – A survey of recent results. *Annals of Mathematics and Artificial Intelligence,* Volume 31, pp. 113-126.

Chua, B. H., 2013. *Integration of Multiple Unmanned Systems in an Urban Search and Rescue Environment,* s.l.: Calhoun.

De Freitas, E. P. et al., 2010. *UAV Relay Network to Support WSN Connectivity.* Moscow, IEEE.

DJI, 2017. *DJI GS PRO.* [Online]
Available at: https://www.dji.com/ground-station-pro/info
[Accessed 20 October 2017].

DroidPlanner Labs, 2017. *Tower.* [Online]
Available at: https://play.google.com/store/apps/details?id=org.droidplanner.android
[Accessed 20 October 2017].

Dronecode Project, 2017. *Dronecode.* [Online]
Available at: https://www.dronecode.org/platform/#ground_control_station
[Accessed 5 October 2017].

Dudek, M. et al., 2013. Hybrid Fuel Cell – Battery System as a Main Power Unit for Small Unmanned Aerial Vehicles (UAV). *International Journal of Electrochemical Science,* Volume 8, pp. 8442-8463.

EASA, 2017. *Civil drones (Unmanned aircraft).* [Online]
Available at: http://www.easa.europa.eu/easa-and-you/civil-drones-rpas
[Accessed 27 September 2017].

Frost, J. R. & Stone, L. D., 2001. *Review of Search Theory: Advances and Applications to Search and Rescue Decision Support,* s.l.: U.S. Coast Guard Research and Development Center.

Gagne, D., 2017. *QGroundControl User Guide.* [Online]
Available at: https://www.gitbook.com/book/donlakeflyer/qgroundcontrol-user-guide/details
[Accessed 19 October 2017].

Gandor, F., Rehak, M. & Skaloud, J., 2015. Photogrammetric Mission Planner for RPAS. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences,* 40(1), pp. 61-65.

Garcia, R. & Barnes, L., 2009. Multi-UAV Simulator Utilizing X-Plane. In: K. P. Valavanis, et al. eds. *Selected papers from the 2nd International Symposium on UAVs.* Reno, Nevada, U.S.A.: Springer, Dordrecht, pp. 393-406.

Gaudiano, P., Shargel, B., Bonabeau, E. & Clough, B. T., 2003. *Control of UAV swarms: What the bugs can teach us.* San Diego, Proceedings of the 2nd AIAA Unmanned Unlimited Systems, Technologies, and Operations Aerospace Conference.

Global UTM Association, 2017. *UAS Traffic Management Architecture.* [Online]
Available at: https://www.gutma.org/docs/Global_UTM_Architecture_V1.pdf
[Accessed 27 October 2017].

Goodrich, M. A. et al., 2008. *Supporting Wilderness Search and Rescue using a Camera-Equipped Mini UAV,* s.l.: Wiley.

Groom, V. et al., 2011. *Responses to Robot Social Roles and Social Role Framing.* Collaboration Technologies and Systems (CTS), IEEE.

Hodgson, J. C. et al., 2017. *Drones count wildlife more accurately and precisely than humans,* s.l.: University of Adelaine.

Hong Kong TAC Industrial Co., Ltd., 2017. *UL Lithium Ion Cylindrical Battery.* [Online]
Available at: http://www.tacbattery.com/sale-9988528-ul-lithium-ion-cylindrical-battery-18650-li-ion-battery-3-6-3-7-v-2600-3400mah-for-flashlights.html
[Accessed 19 October 2017].

Hooper, P., 2005. *Stepped piston engines for multi-fuel UAV application.* Bristol, Institution of Mechanical Engineers.

Joint Authorities for Rulemaking on Unmanned Systems, 2016. *Regulations.* [Online]
Available at: http://jarus-rpas.org/regulations
[Accessed 21 September 2017].

Jurecka, M. & Niedzielski, T., 2017. A procedure for delineating a search region in the UAV-based SAR activities. *GEOMATICS, NATURAL HAZARDS AND RISK,* 8(1), p. 53–72.

Keane, J. F., 2013. A Brief History of Early Unmanned Aircraft. *Johns Hopkins APL Technical Digest,* 32(3), pp. 558-571.

Kohls, M., 2016. *Expected Coverage of Random Walk Mobility Algorithm,* s.l.: s.n.

Kou, K.-H., Geng, B.-L. & Zhang, F.-X., 2017. *Collaborative Strategy of Multi-UAV for Searching Sea Area,* s.l.: International Conference on Electrical Engineering and Automation Control.

Krupnik, A., 2012. *Rocket-power SkyFun.* [Online]
Available at: http://diydrones.com/profiles/blogs/rocket-power-skyfun
[Accessed 24 October 2017].

Leitner, J., 2009. *Multi-Robot Formations for Area Coverage in Space Applications.* s.l.:s.n.

Lin, L. & Goodrich, M. A., 2010. A Bayesian approach to modeling lost person behaviors based on terrain features in Wilderness Search and Rescue. *Comput Math Organ Theory,* Issue 16, pp. 300-323.

Lin, R. L., 2009. *UAV Intelligent Path Planning for Wilderness Search and Rescue,* Provo: Brigham Young University.

Li, Y., Er, M. J. & Wang, X., 2011. Coverage path planning for UAVs based on enhanced exact cellular decomposition method. *Mechatronics,* 21(5), pp. 876-885.

Lowc Technology Co., Ltd, 2017. *LED Lithium polymer battery 1400mAh.* [Online]
Available at: http://www.lowctech.com/html_products/LED-Lithium-polymer-battery-1400mAh-36.html
[Accessed 19 October 2017].

Luke, T. & Uchizono, N., 2011. *Wirelessly Operated PA Loudspeaker System for RMAX Search-and-Rescue UAV,* s.l.: s.n.

Lukonis, D., 2014. *Solar Drone Experiments - How much more battery life do you get by adding solar panels to your quad?.* [Online]
Available at: http://diydrones.com/profiles/blogs/solar-drone-experiments-how-much-more-battery-life-do-you-get-by
[Accessed 19 October 2017].

Mackay, R., 2016. *ArduCopter Release Notes.* [Online]
Available at: https://github.com/ArduPilot/ardupilot/blob/master/ArduCopter/ReleaseNotes.txt
[Accessed 29 July 2017].

Mallesh, B. S., Lokesha, H., Veena, S. & Jayantkumar, R. A., 2015. *Design of Speech Based Ground Control Station for Controlling the Micro Air Vehicles.* Kochi, India, IEEE.

Maza, I. & Ollero, A., 2004. *Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms.* Seville, University of Seville.

McCray, G., 2015. *Batteries For UAV.* [Online]
Available at: http://www.dronesarefun.com/BatteriesForUAV.html
[Accessed 19 October 2017].

McNabb, M., 2017. *Drone Integration into European Airspace Ready for "Immediate Deployment".* [Online]
Available at: https://dronelife.com/2017/09/14/drone-integration-into-european-airspace-takes-a-big-step-forward/
[Accessed 27 October 2017].

Meier, L., 2009. *MAVLink -- Micro Air Vehicle Message Marshalling Library.* [Online]
Available at: https://github.com/mavlink/mavlink/
[Accessed 30 July 2017].

Meier, L., 2017. *MAVLink Mission Interface.* [Online]
Available at: http://www.mavlink.org/mavlink/mission_interface
[Accessed 9 October 2017].

Mirzaei, M. et al., 2011. *Cooperative Multi-Vehicle Search and Coverage Problem in Uncertain Environments.* Orlando, 50th IEEE Conference on Decision and Control and European Control Conference.

NASA, 2017. *Unmanned Aircraft System (UAS) Traffic Management (UTM).* [Online]
Available at: https://www.utm.arc.nasa.gov/
[Accessed 25 October 2017].

Nebiker, S., Annen, A., Scherrer, M. & Oesch, D., 2008. A light-weight multispectral sensor for micro UAV—Opportunities for very high resolution airborne remote sensing. *The international archives of the photogrammetry, remote sensing and spatial information sciences,* Volume 37, pp. 1193-1199.

Nedjati, A., Izbirak, G., Vizvari, B. & Arkat, J., 2016. Complete Coverage Path Planning for a Multi-UAV Response System in Post-Earthquake Assessment. *Robotics,* 5(26).

Newaz, A. A. R., Jeong, S. & Chong, N. Y., 2016. *Fast Radioactive Hotspot Localization Using a UAV.* s.l., IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots.

Nex, F. & Remondino, F., 2014. UAV for 3D mapping applications a review. *Applied Geomatics,* 6(1), pp. 1-15.

Oborne, M., 2016. *Mission Planner.* [Online]
Available at: http://ardupilot.org/planner/
[Accessed 30 July 2017].

Oettershagen, P. et al., 2016. Long-Endurance Sensing and Mapping Using a Hand-Launchable Solar-Powered UAV. *Field and Service Robotics,* Volume 113, pp. 441-454.

Pastor, E., Lopez, J. & Royo, P., 2007. UAV Payload and Mission Control Hardware/Software Architecture. *IEEE Aerospace and Electronic Systems Magazine,* 22(6), pp. 3-8.

Perkins, D., Roberts, P. & Feeney, G., 2003. *Missing Person Behaviour - An Aid to the Search Manager.* 1st ed. s.l.:Northumberland National Park SRT, Centre for Search Research.

Popovic, M. et al., 2016. *Online Informative Path Planning for Active Classification on UAVs,* Zürich: Autonomous Systems Lab.

Quadcopter-Addiction.com, 2017. *Skyfront Sets World Record For Drones With 4 Hour And 34 Minute Flight.* [Online]
Available at: http://quadcopter-addiction.com/2017/09/14/skyfront-sets-world-record-drones-4-hour-34-minute-flight/
[Accessed 24 October 2017].

Radioman, 2008. *greatmaps.* [Online]
Available at: https://github.com/radioman/greatmaps
[Accessed 30 July 2017].

Rosalie, M., Danoy, G., Chaumette, S. & Bouvry, P., 2016. *UAV Multilevel Swarms for Situation Management,* s.l.: HAL.

Sánchez-García, J. et al., 2016. An Intelligent Strategy for Tactical Movements of UAVs in Disaster Scenarios. *International Journal of Distributed Sensor Networks,* 12(3).

Scott, K. K., 2016. *Occlusion-Aware Sensing and Communication in Unmanned Aerial Vehicle (UAV) Networks,* s.l.: University of Cincinnati.

Skinner, T., 2016. *Telecoms.com.* [Online]
Available at: http://telecoms.com/472851/nokia-announces-portable-ran-breakthrough-so-do-vodafone-and-ericsson/
[Accessed 25 September 2017].

SPH Engineering, 2017. *Drone Dance Controller.* [Online]
Available at: https://www.ugcs.com/en/page/ddc
[Accessed 20 October 2017].

SPH Engineering, 2017. *Mission planner | Drone control.* [Online]
Available at: https://www.ugcs.com/en/ugcs_features_applications
[Accessed 20 October 2017].

Sullivan, B., 2016. *Capabilities of Flight Controllers for UAV Group Flight,* s.l.: Lehigh University.

Sundqvist, L., 2015. *Cellular Controlled Drone Experiment: Evaluation of Network Requirements,* Espoo: Aalto University School of Electrical Engineering.

Suzuki, K. A. O., Filho, P. K. & Morrison, J. R., 2011. *Automatic Battery Replacement System for UAVs: Analysis and Design.* s.l., Springer Science+Business Media B.V..

Szafir, D., Mutlu, B. & Fong, T., 2017. Designing planning and control interfaces to support user collaboration with flying robots. *The International Journal of Robotics Research,* pp. 1-29.

Tao, P., 2016. *Design, prototyping and autonomous control of gasoline-engine variable-pitch quadcopter,* s.l.: National University of Singapore.

Tridgell, A., Barker, P. & Dade, S., 2016. *MAVProxy.* [Online]
Available at: http://ardupilot.github.io/MAVProxy/html/index.html
[Accessed 19 October 2017].

Wang, J., Schluntz, E., Otis, B. & Deyle, T., 2015. *A New Vision for Smart Objects and the Internet of Things: Mobile Robots and Long-Range UHF RFID Sensor Tags,* s.l.: arXiv preprint arXiv:1507.02373.

VC Technology Ltd, 2017. *Litchi.* [Online]
Available at: https://flylitchi.com/
[Accessed 20 October 2017].

Wollan, H., 2004. *Incorporating Heuristically Generated Search Patterns in Search and Rescue,* s.l.: University of Edinburgh.

Zarco-Tejada, P. J. et al., 2009. Imaging chlorophyll fluorescence with an airborne narrow-band multispectral camera for vegetation stress detection. *Remote Sensing of Environment,* 113(6), pp. 1262-1275.

Zhang, M., Song, J., Huang, L. & Zhang, C., 2017. *Distributed Cooperative Search with Collision Avoidance for a Team of Unmanned Aerial Vehicles Using Gradient Optimization,* s.l.: ASCE.

Zhu, B. et al., 2017. *A survey on recent progress in control of swarm systems,* s.l.: Science China.

Appendix A – Mission footage